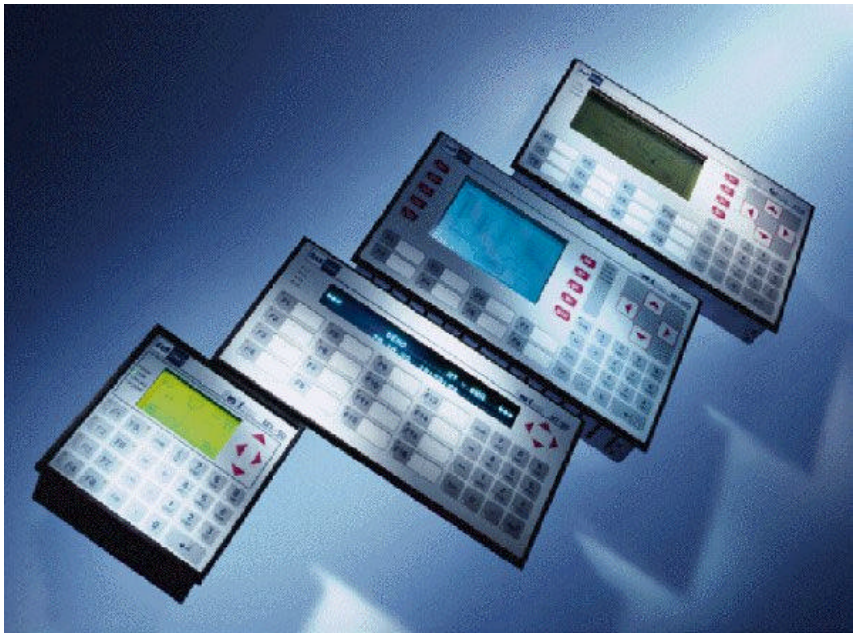
SAE-STAHL GMBH
Im Gewerbegebiet Pesch 14
D-50767 Köln
Tel.: +49-221-59808-200
Fax: +49-221-59808-60
http://www.sae-stahl.de
E-mail:office@sae-stahl.de

Manual
EP-Modbus/ EP-COM OPC Server



Manual:

EP-Modbus/ EP-COM OPC Server

Article-No.:

Version: 1.00 (Prerelease)

Edition: 24.01.01

Publisher and copyright by

SAE-STAHL GMBH Köln
Im Gewerbegebiet Pesch 14
D-50767 Köln

Phone	(exch.)	+49/(0)221/ 5 98 08	- 200
	(hotline)		- 59
Facsimile			- 60

- All rights reserved.
- This document may not be reproduced in whole or in part except with the written approval of the publisher.
- Technical changes may be made at any time without prior announcement.

Every care has been exercised in compiling this Manual. SAE-STAHL GMBH cannot be held responsible for any errors or omissions contained in this Manual.

IBM is a registered trademark of IBM-Corp.

MS DOS, WINDOWS 95, WINDOWS 98, WINDOWS NT, MICROSOFT ACCESS are registered trademarks of MICROSOFT Corporation

Cologne, dated 24.01.01

Contents

1. Introduction 6

1.1. Readers guide 6
1.2. Product introduction 7
1.2.1. What is EP-Modbus? 7
1.2.2. What is EP-COM? 7

2. Before You start 8

2.1. Delivery 8
2.2. Software license agreement 9

3. Installation 12

3.1. The OPC server packages content 12
3.2. System requirements 12
3.3. How to install 12
3.4. Deinstallation 17

4. Getting started 19

4.1. Example on the CD 19
4.2. The SPSplusWIN project 20
4.3. The EP-Modbus configuration database 22
4.4. The OPC DA Custom Client 26

5. Configuration procedure - example application 28

5.1. Application specification 28
5.2. MT300 application project 29
5.3. EPModbus configuration database 31
5.4. EPCom configuration database 32
5.4.1. Device 32
5.4.2. Data Items 32
5.5. The OPC client 34

6. Introduction to OPC servers 35

6.1. OLE for Process Control (OPC) 35
6.1.1. OPC Fundamentals 35
6.1.2. OPC Servers 35

6.1.3. OPC Server Interfaces 36
6.1.4. OPC DataAccess Server 37
6.2. References 38

7. Exicom/ProVicom OPC Server 39

7.1. Exicom/ProVicom OPC server 39
7.2. Implemented OPC interfaces 40
7.3. Exicom/ProVicom OPC server architecture 41
7.4. Address space and configuration database 42
7.5. EPModbus OPC server 43
7.5.1. Features 43
7.5.2. Address space hierarchy 44
7.5.3. How the server works 45
7.6. EPCom OPC server 46
7.6.1. Purpose 46
7.6.2. Address space hierarchy 47

8. Exicom/ProVicom OPC server configurators 48

8.1. User interface 48
8.1.1. Views layout 48
8.1.2. File menu 49
8.1.3. Edit menu 50
8.1.4. View menu 51
8.1.5. Go menu (browsing the data) 51
8.1.6. Tools menu 52
8.1.7. Popup menus 52
8.2. Configuration of the EPModbus address space 53
8.2.1. Ports 53
8.2.2. Devices 54
8.2.3. Folders 55
8.2.4. Data Items 55
8.2.5. Device Parameters 57
8.2.6. Conversions 58
8.2.7. Simulation Signals 59
8.3. Configuration of the EPCom address space 60
8.3.1. Device 60
8.3.2. Data Items 62

9. Exicom/ProVicom OPC server administrators 67

10. Configuration using SPSPlusWin 68

10.1. Step by step procedure 68
10.2. The hierarchy of the SPSPlusWin projects 70

10.2.1. The root database 70
10.2.2. The leaf database 70
10.2.3. The generated configuration databases 71

11. OPC Clients 72

11.1. OPC Custom client 72
11.1.1. How to connect to a server? 73
11.1.2. How to display server status information? 73
11.1.3. How to change group parameters 74
11.1.4. How to add a new item to the server group 75
11.1.5. How to write a value to the item 75
11.2. OPC DA Automation client 76
11.2.1. How to run the OPCExampleClient? 76
11.2.2. Sheet1 – List of OPC DA servers 77
11.2.3. Sheet2 – Server status 77
11.2.4. Sheet3 – Server items 78
11.2.5. Sheet4 – Viewing and editing the product list 79

1. Introduction

1.1. Readers guide

Chapter 1 shortly intoduces the products EP-Modbus and EP-COM

Chapter 2 gives important information you should notice before installing the software.

Chapter 3 leads through the installation and deinstallation procedures. Additionally, the software modules of the program package and the system requirements are listed.

Chapter 4 describes how to

Chapter 5 illustrates on a simple example the configuration procedure. It describes step by step how to configure the Exicom/ProVicom OPC server.

Chapter 6 gives some background information about the OPC terminology. It describes the general structure of an OPC Data Access server.

Chapter 7 describes the architecture of the Exicom/ProVicom OPC server. It describes the features, functions and the hierarchy of the address space of the server.

Chapter 8 provides detailed information about configurators - the programs for configuring of the server. The detailed description of the address space of the EPCom OPC server and of the EPModbus OPC server is provided in this chapter.

Chapter 9 describes the administrators – the programs for administration of the configuration databases of the Exicom/ProVicom OPC server.

Chapter 10 how to generate the configuration databases of the EPModbus and EPCom OPC servers in the environment of the SPSPlusWin engineering tool.

Chapter 11 describes two OPC client programs. One of the clients uses the custom OPC interface and it was written in C++ language. The second client uses the OPC Automation Wrapper and was written in Visual Basic for Applications language (VBA).

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

1.2. Product introduction

The Exicom/ProVicom OPC servers EP-Modbus and EP-COM are out-process OPC DataAccess servers. They enable OPC clients to communicate with SAE-STAHL Exicom/ProVicom operator terminals.

1.2.1. What is EP-Modbus?

EPModbus OPC server implements MODBUS communication protocol. This server enables to access each variable defined in the application of the Exicom/ProVicom device.

- Supports the link of devices to serial interfaces (RS-232, RS-422)
- Support of COM 1-4
- Data exchange via Modbus RTU
- Bus Adressing up to 31 stations (dependent on the volume of data)
- Handling block for MS Windows
- Applicable for soft PLC and simple connections

1.2.2. What is EP-COM?

EPCom OPC server controls the underlying EPModbus OPC server according to the user-defined conditions. For example it enables to transfer data only from variables defined on the current page of the device or only if a key was pressed. It also enables to create records of more variables.

- Uses the EP Modbus
- Reduces data exchange by event control
  - Page query after depressing the key (formulas)
  - Page refresh after modifications
  - Creation of data structures
- takesover datenmanagement functions
- application for data base controlledsystems
  - ACCESS
  - Excel
  - Oracle

The EP COM software package includes EP Modbus.

2. Before You start

2.1. Delivery

The delivery contents:
- Program CD
- This manual

2.2. Software license agreement

Between **SAE-STAHL GMBH**
Im Gewerbegebiet Pesch 14
D-50767 Köln

- herein referred to as Seller -

and

- herein referred to as Buyer

the following contract is made.

§1 Object of contract

(1) Seller sells to Buyer the EDP program package described in this Manual, together with the Manual.
(2) The program is stored on machine-readable medium.
(3) The program and the User's Manual are copyrighted pursuant to copyright act of 25.06.198.

§2 Conclusion of contract

Buyer accepts the offer submitted by Seller in this contract form.

§3 Use

The right to use the program package is granted for one data-processing device in each case. Buyer may select the device on which he wants to use the program package.
Buyer is entitled to save the program repeatedly in the device in which he intends to use it from time to time or permanently for the purpose of executing the program and make copies of the program as may be required for its use. This right also includes the right to duplicate the program for use at the same time by several data-processing devices within one facility. Duplication of the manual is not allowed except with the written permission of the copyright holder.

§4 Passing on of the program package and creation of new user rights

(1) Buyer may pass on the program package in its original form for use by an after user. This right does not exist for passing on of any modified or edited versions or copies of that version, including partial copies.
(2) Buyer may agree to grant the after user own user right. That right shall be of the same nature and scope as the Buyer's continuing right pursuant to §3.

§5 Passing on by after user

The provisions of §3 shall apply *mutatis mutandis* if the program package is passed on by the after user to another after user.

§6 Other rights

Any right to use or exploit the program package beyond the right granted under the preceding paragraphs, especially those under §§15 and 34 copyright act, are reserved.

§7 Warranty

(1) It should be noted that it is not possible to develop data processing programs free of fault in all later application environments. Seller represents that the program as described in the manual effective at the time at which it is delivered to Buyer can be used and has the attributes assured therein. Any insignificant impairment of the usability does not limit usability generally.

(2) Seller warrants that the original program has properly been recorded on a verified volume.

(3) If it shall be found that the program package is not usable within the meaning of paragraph 1 or is defective within the meaning of paragraph 2, Buyer may require within a warranty period of six months beginning from the delivery of the program package that Seller receive back the delivered program package and supply instead a new program packages of the same type and description. If it shall be found that the program package supplied in replacement of the unusable or defective program package is also not usable within the meaning of paragraph 1 or is defective within the meaning of paragraph 2 and provided the entitled person fails to restore the usefulness with reasonable effort and within reasonable time, then and in such case Buyer or user, as the case may be, shall be entitled at its option either to a reduction of the purchase price or return of the program package and reimbursement of the purchase price.

(4) Paragraphs 1-3 also apply to further users as intended by §3. Warranty begins at the time defined in §3, not at the time the program package is delivered to the further user.

(5) There is no other warranty except that described above. In particular, no warranty is made that the program package meets the specific Buyer requirements. Buyer shall have sole responsibility for the selection, installation and use as well as for the results intended by these. Further, no warranty is made for any modified or revised versions of the program.

§8 Liability

(1) Seller is liable for damage for which he or the entitled person may be responsible up to an amount equal to the purchase price. No liability is undertaken for lost profit, unrealized savings, indirect damage or consequential damage of any kind.

(2) The above limited liability does not exist for warranted features or damage due to intent or gross neglect.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

§9 Comprehensive retention of title

(1) The program package as delivered remains Seller's property pursuant to § 454 BGB German Civil Code until all claims from business transactions, whether current or future, have been satisfied in full. Ownership as above exists even if any claim should be made part of a current invoice and a balance is struck and accepted.

(2) Buyer hereby assigns any claim from the resale or otherwise of the goods title to which is retained (including but not limited to any balance due under current account) and any other rights to Seller as surety. In the event of Seller passing on the program package within the meaning of §84, Seller is obliged to agree retention of title with the subsequent user or users. Title in the goods shall be deemed to be retained in favor of Seller until full payment of the related claim is made.

(3) Buyer is irrevocably entitled at any time and in his name to collect any claim assigned to Seller as above, the amount of such claim to be credited to Seller's account. Orderer is obliged to inform Seller on the latter's request of the amount of his claim, the reason why it exists and the name of the third debtor.

(4) No claim may be pawned or pledged as security. If Buyer's business should be seized by way of execution or bankruptcy proceedings opened or he should compound with his debtors, Buyer is obliged to refer to the existence of Seller's retained title and inform Seller accordingly without delay.

(5) On Buyer's request, Seller undertakes to release any security furnished to him to the extent to which the value of such security exceeds the value of the secured claim by more than 20%.

§ 11 General provisions

(1) The place of performance of all obligations under this contract is Cologne, Germany.

(2) The jurisdiction for all differences arising out of this contract shall also be Cologne, Germany.

(3) All amendments and modifications of this contract in order to be effective shall be made in writing and shall make express reference to this contract and require the signature of both parties.

(4) If any of the provisions in this contract shall be or become ineffective, this shall not affect the other provisions. The parties hereto will replace an ineffective provision by an effective provision as closely as possible resembling the economic intent of the replaced provision.

3. Installation

3.1. The OPC server packages content

The software package includes:

- EP-Modbus Server
- EP-COM Server
- EP-Modbus Configurator
- EP-Modbus Administrator
- EP-COM Configurator
- EP-COM Administrator
- OPC DA Automation Client
- OPC DA Custom Client
- Readme file
- Uninstall program

3.2. System requirements

- Microsoft Windows NT 4.0 or Windows 95 (with DCOM installed) or Windows 98
- 64 MB RAM
- 30 MB free space on hard disk

We recommend that you use Windows NT 4.0 with 128 MB RAM.

3.3. How to install

The start menu is automatically launched when the CD is inserted in the drive. When the autostart function is disabled you can start it manually by calling the program START.EXE.

In the menu INSTALL you can select the module you want to install. Select the module Exicom-ProVicom OPC Server to install the OPC Server. You can also install the Microsoft Data Access Components you need for working with the configuration databases.

When you have installed Microsoft Access the Data Access Components are installed before.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

Figure 1– The start menu with the modules to install

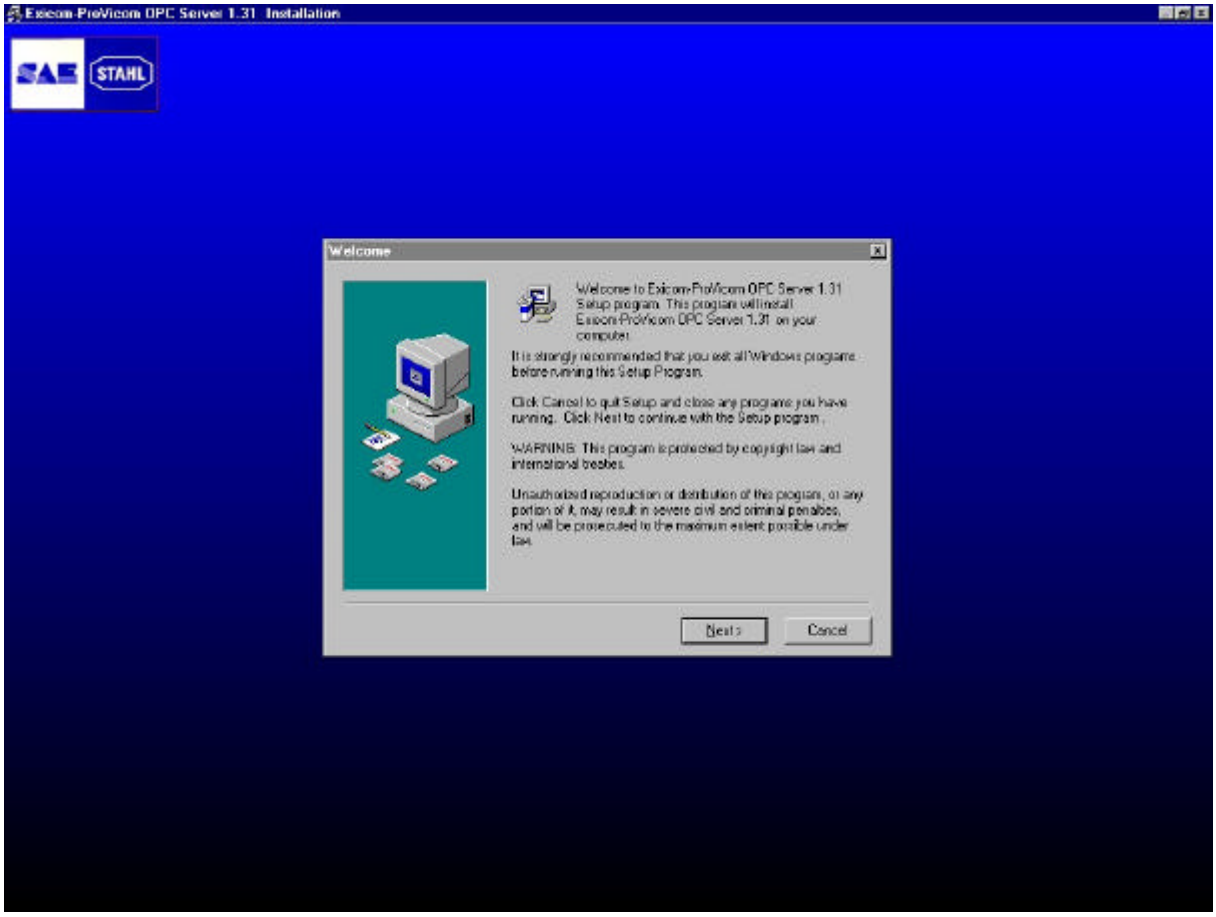

Exicom-ProVicom OPC Server 1.31 Installation
SAE
STAHL
Welcome
Welcome to Exicom-ProVicom OPC Server 1.31 Setup program. This program will install Exicom-ProVicom OPC Server 1.31 on your computer.
It is strongly recommended that you exit all Windows programs before running this Setup Program.
Click Cancel to quit Setup and close any programs you have running. Click Next to continue with the Setup program.
WARNING: This program is protected by copyright law and international treaties.
Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.
Next >
Cancel

Figure 2 – The install program for Exicom-ProVicom OPC Server

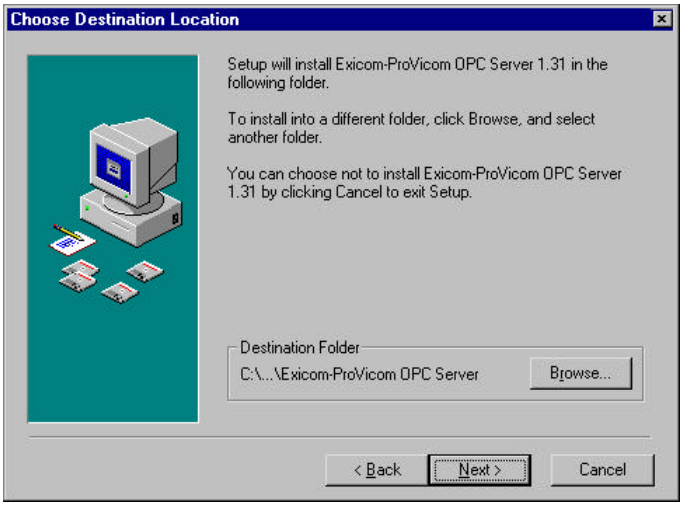

Choose Destination Location
Setup will install Exicom-ProVicom OPC Server 1.31 in the following folder.
To install into a different folder, click Browse, and select another folder.
You can choose not to install Exicom-ProVicom OPC Server 1.31 by clicking Cancel to exit Setup.
Destination Folder
C:\...\Exicom-ProVicom OPC Server
Browse...
< Back
Next >
Cancel

Figure 3 – Choose destination location

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

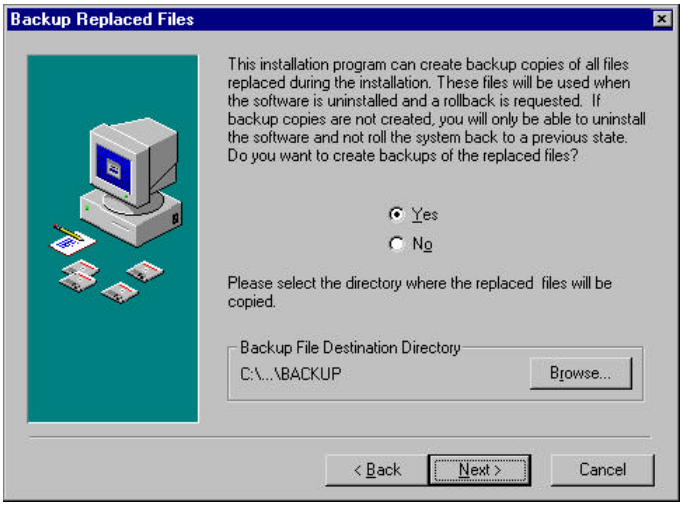

Backup Replaced Files
This installation program can create backup copies of all files replaced during the installation. These files will be used when the software is uninstalled and a rollback is requested. If backup copies are not created, you will only be able to uninstall the software and not roll the system back to a previous state. Do you want to create backups of the replaced files?
Yes
No
Please select the directory where the replaced files will be copied.
Backup File Destination Directory
C:\...\BACKUP
Browse...
< Back
Next >
Cancel

Figure 4 – Backup the replaced files

It is recommended that you create backups when you want to be able to roll the system back to the previous state when you uninstall the OPC server.

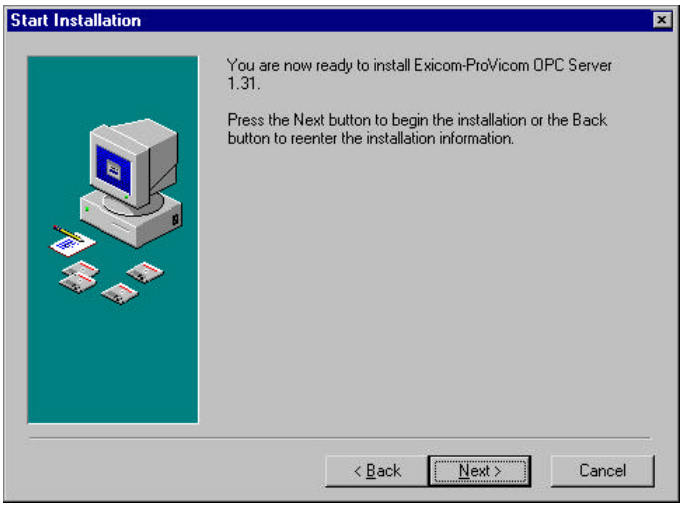

Start Installation
You are now ready to install Exicom-ProVicom OPC Server 1.31.
Press the Next button to begin the installation or the Back button to reenter the installation information.
< Back
Next >
Cancel

Figure 5 – Start installation

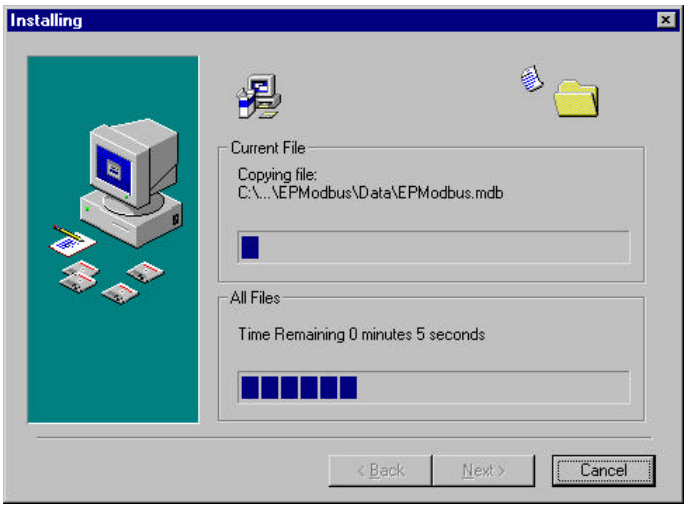


Figure 6 – Installing the OPC Server

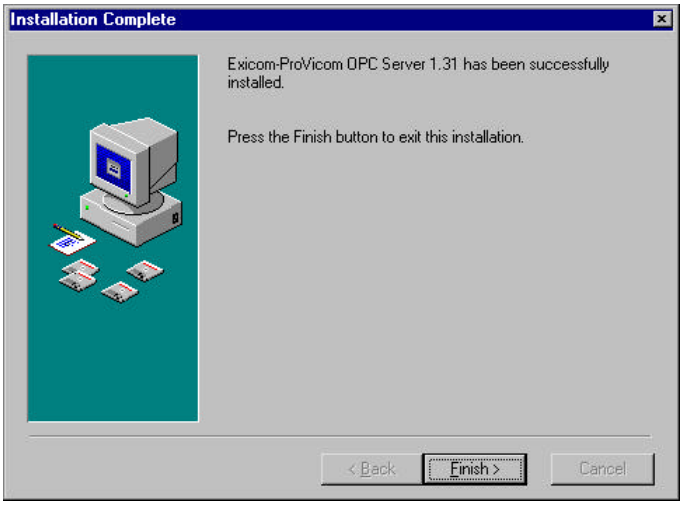

Installation Complete
Exicom-ProVicom OPC Server 1.31 has been successfully installed.
Press the Finish button to exit this installation.
< Back
Finish >
Cancel

Figure 7 – Installation completed

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

3.4. Deinstallation

You can deinstall the Exicom-ProVicom OPC-Server with the uninstall program you find in the start menu.

Please press **Start➔Program➔SAE-STAHL Exicom-ProVicom OPC-Server➔Uninstall**.

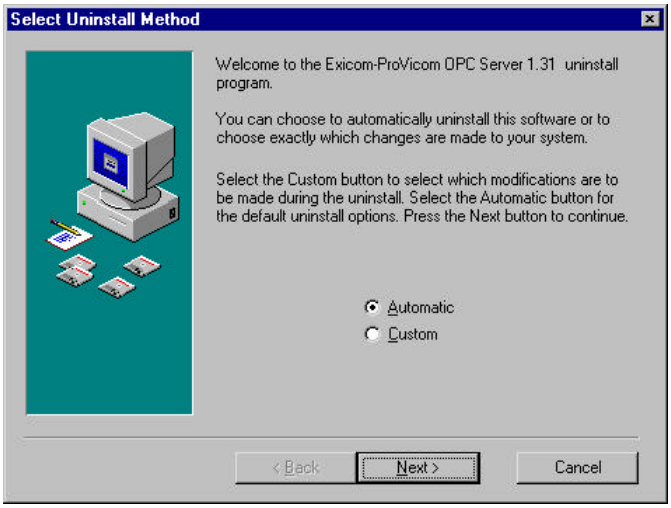

Select Uninstall Method
Welcome to the Exicom-ProVicom OPC Server 1.31 uninstall program.
You can choose to automatically uninstall this software or to choose exactly which changes are made to your system.
Select the Custom button to select which modifications are to be made during the uninstall. Select the Automatic button for the default uninstall options. Press the Next button to continue.
Automatic
Custom
< Back
Next >
Cancel

Figure 1 – Select the uninstall method

It is recommended that you select the automatic uninstall to roll back the system correctly

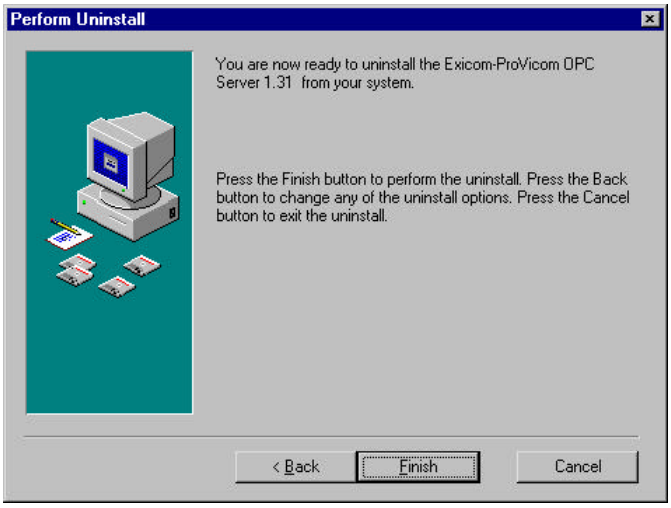

Perform Uninstall
You are now ready to uninstall the Exicom-ProVicom OPC Server 1.31 from your system.
Press the Finish button to perform the uninstall. Press the Back button to change any of the uninstall options. Press the Cancel button to exit the uninstall.
< Back
Finish
Cancel

Figure 2 – Perform uninstall

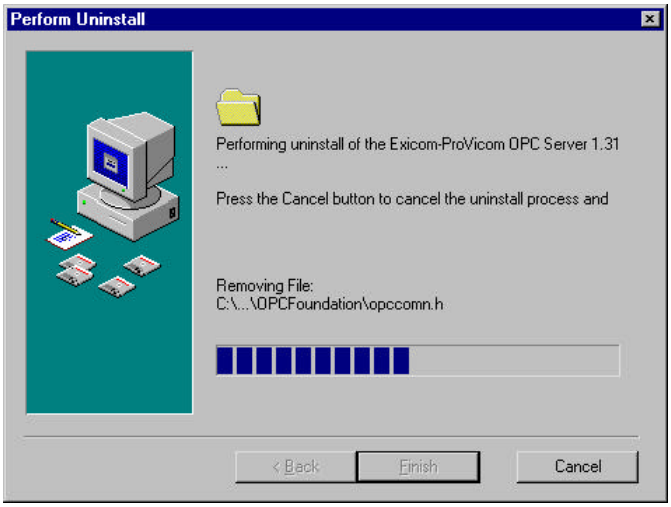


Figure 3 – Uninstalling the OPC Server

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

4. Getting started

4.1. Example on the CD

In this example the terminal MT-45 is used. You find this example on the CD in the folder:
\Examples\OPC_MT-45

The folder containes following files:

- **OPC_MT-45.zip** Archive of the SPSplusWIN project.
- **EPModbusMT45.mdb** Configuration database of the EP-Modbus Server
- **EPModbusMT45.opc** OPCTestClient File of the OPC DA Custom Client

You can use this example with other terminals, when you change the terminal specific parts in this project files. For further information about the terminals and SPSplusWIN please refer to the specific manuals.

4.2. The SPSplusWIN project

To open the demo project please load the archive in SPSplusWIN with the command **File➔Load Archive**. The project has one page with three input/output fields, which. have references to variables with process connections.

For example: The integer variable has the name IntVarSend with the address 100.

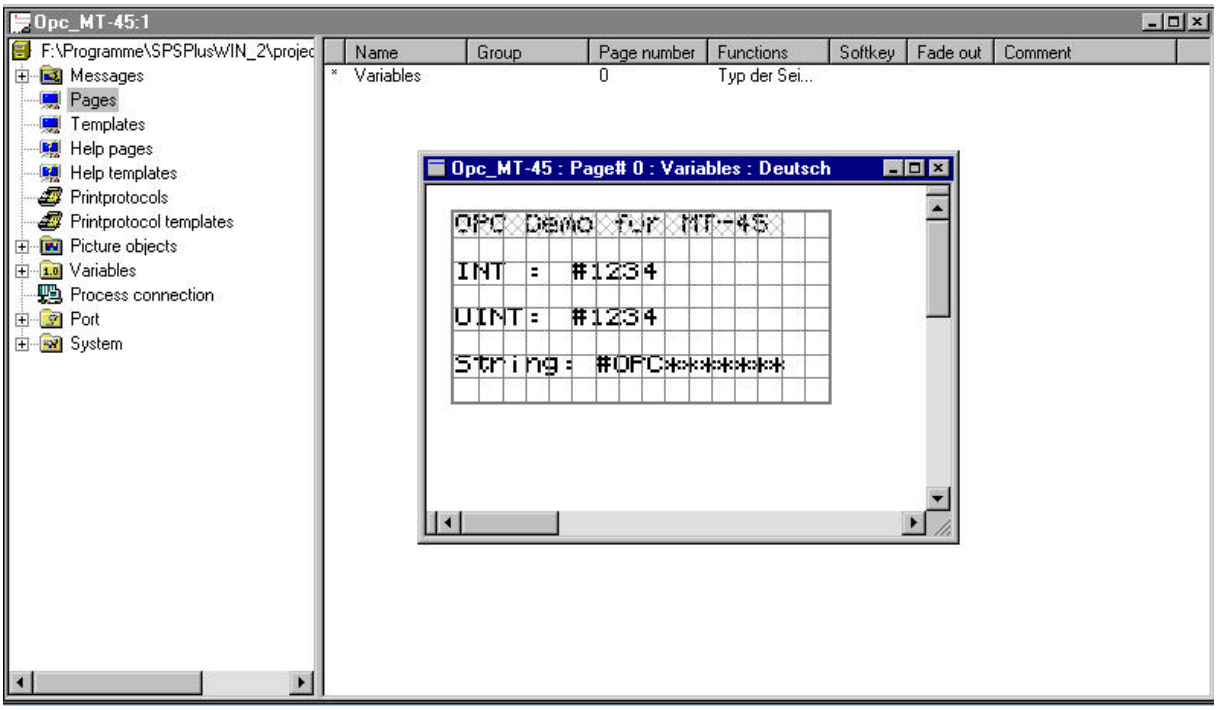


Figure 1 – The page variables

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

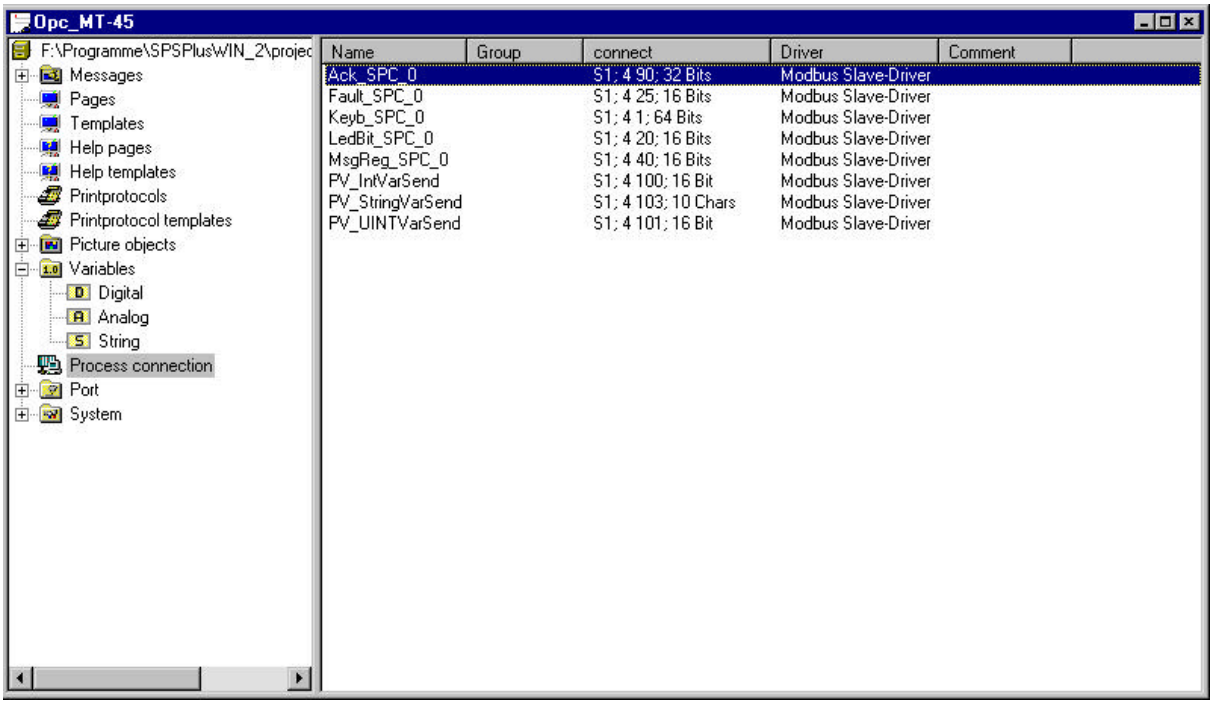


Figure 2 – The process connections of the variables

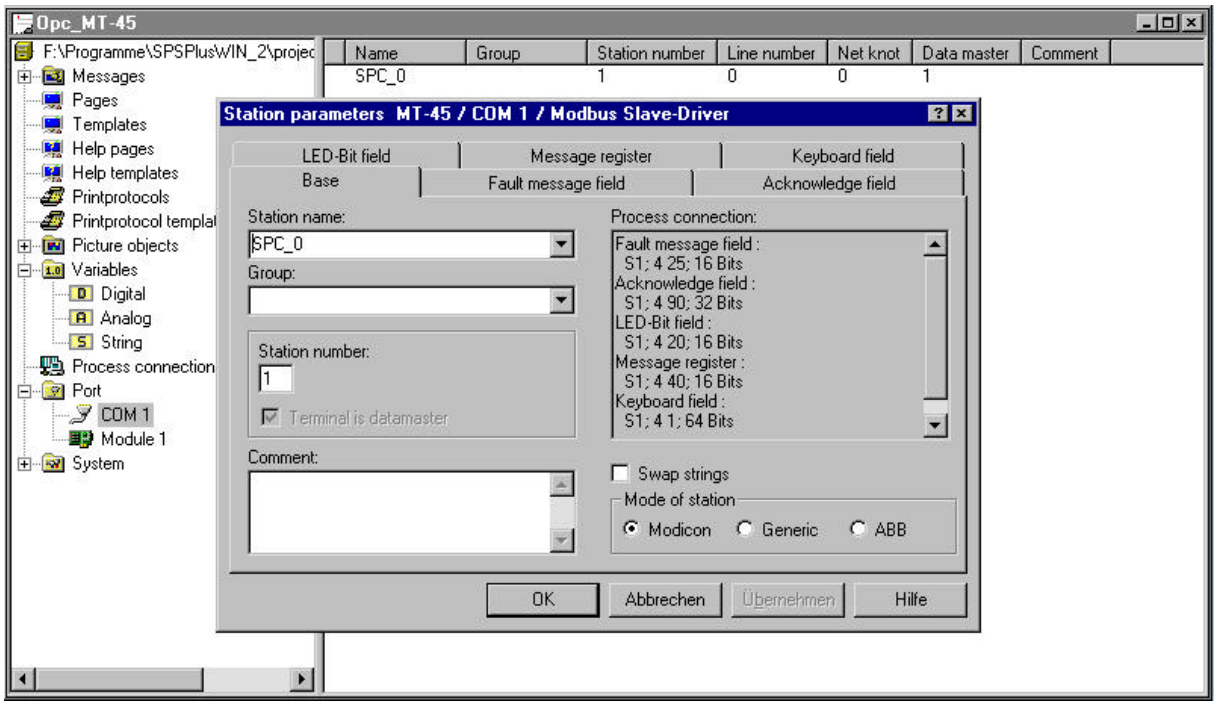


Figure 3 – The station parameters

In the sheet „station parameters“ are the terminal functions defined. You find more information about it in the protocol driver manual for Modbus in the chapter **„Function description of PLC-Mode“**.

Please download this project in the terminal with the command **Project➔Transfer**.

4.3. The EP-Modbus configuration database

The EP-Modbus Configurator can be opened with the start menu.
Please press **Start➔Program➔SAE-STAHL Exicom-ProVicom OPC-Server ➔EPModbus Configurator.**

Open the example database **EPModbusMT45.mdb** and set it as active database with the commands **File➔Open** and **File➔Make Active**.

On the left side of the configurator is a tree view with the address space of the OPC Server. In this address space you can define the ports, the devices and the data items.

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

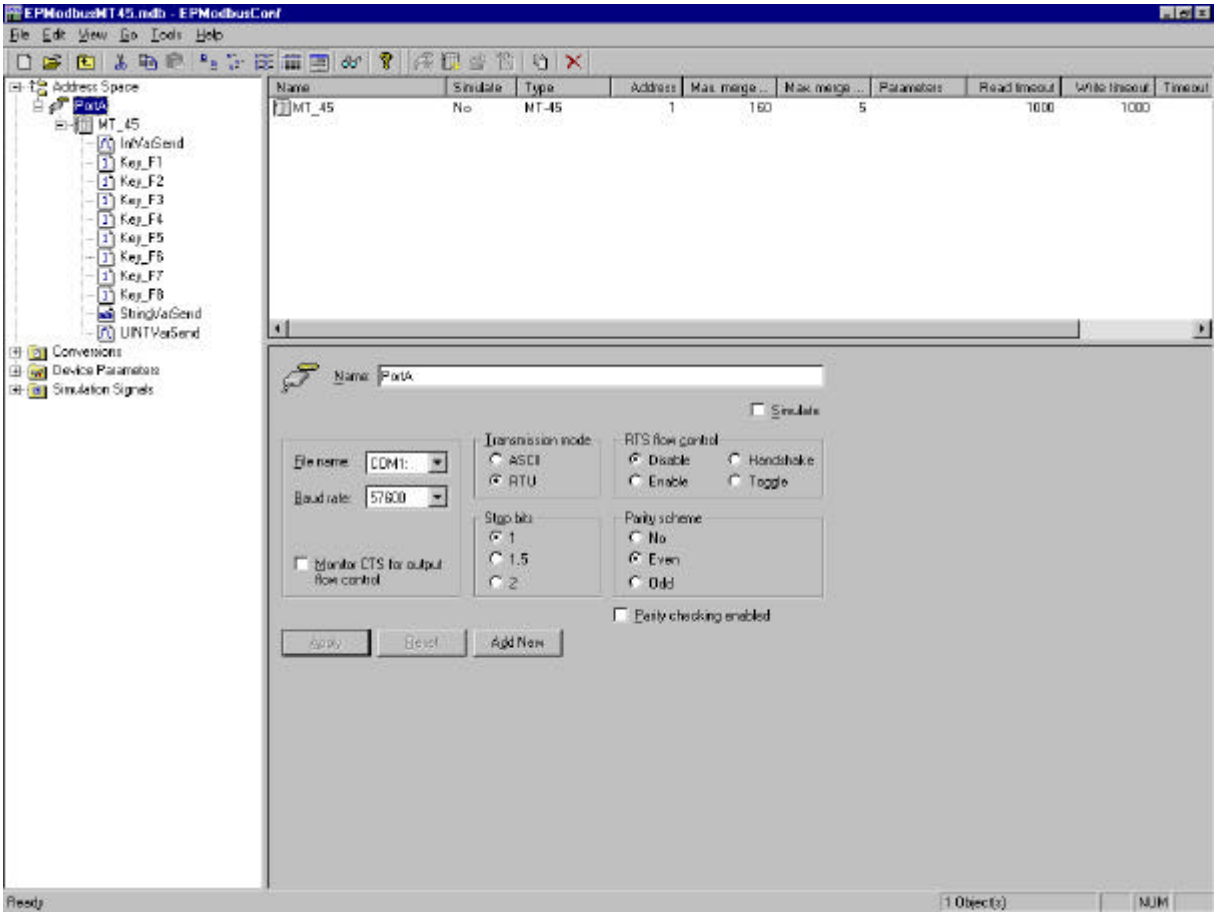


Figure 1 – The EP Modbus Configurator with the port parameters

In the branch **Port** you define the parameters of the communication port of the OPC Server. The parameters must be equal with the communication parameters of the terminal.

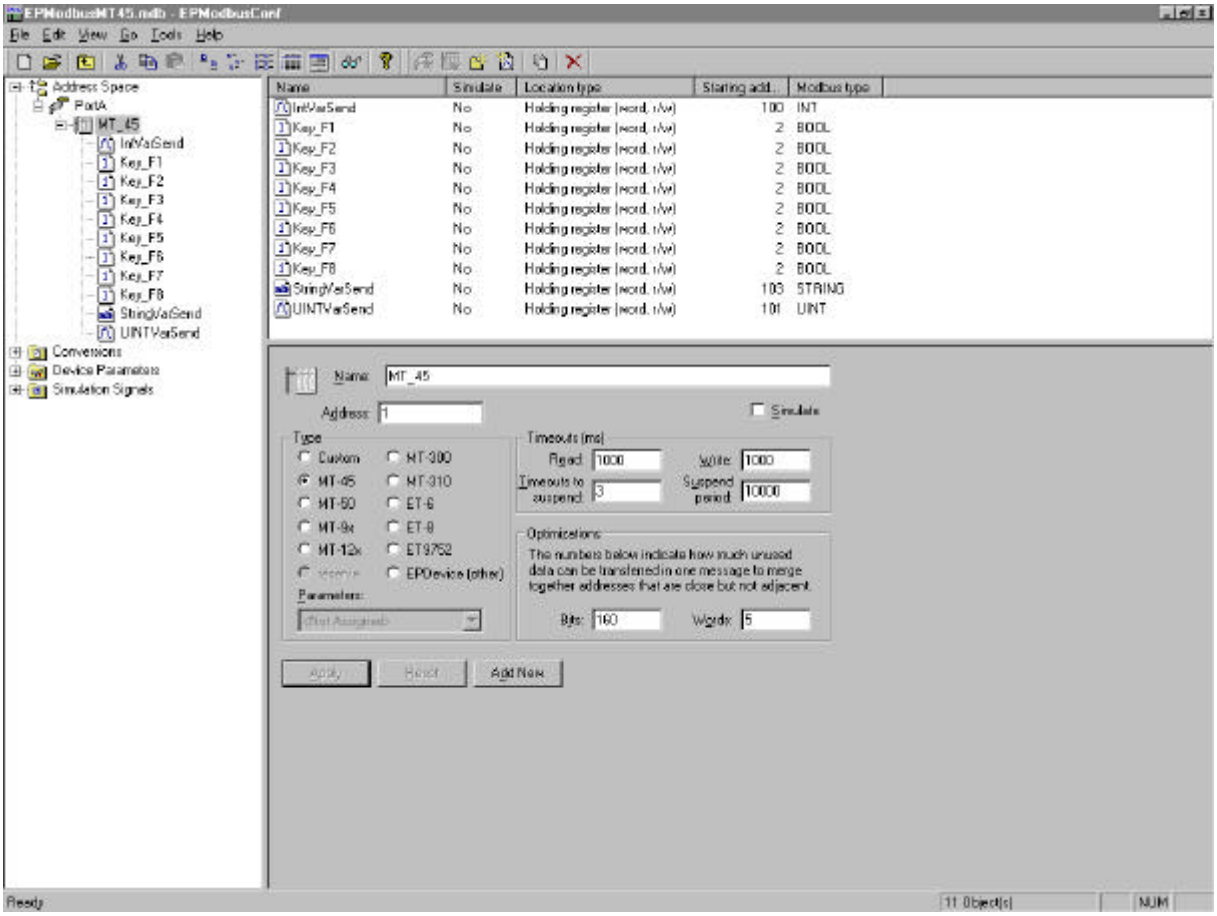


Figure 2 – The EP Modbus Configurator with the device parameters

In the branch **Device** you define the parameters of the terminal on this communication line. The address must be equal with the station number of the terminal.

When you close the database changes are saved automatically.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

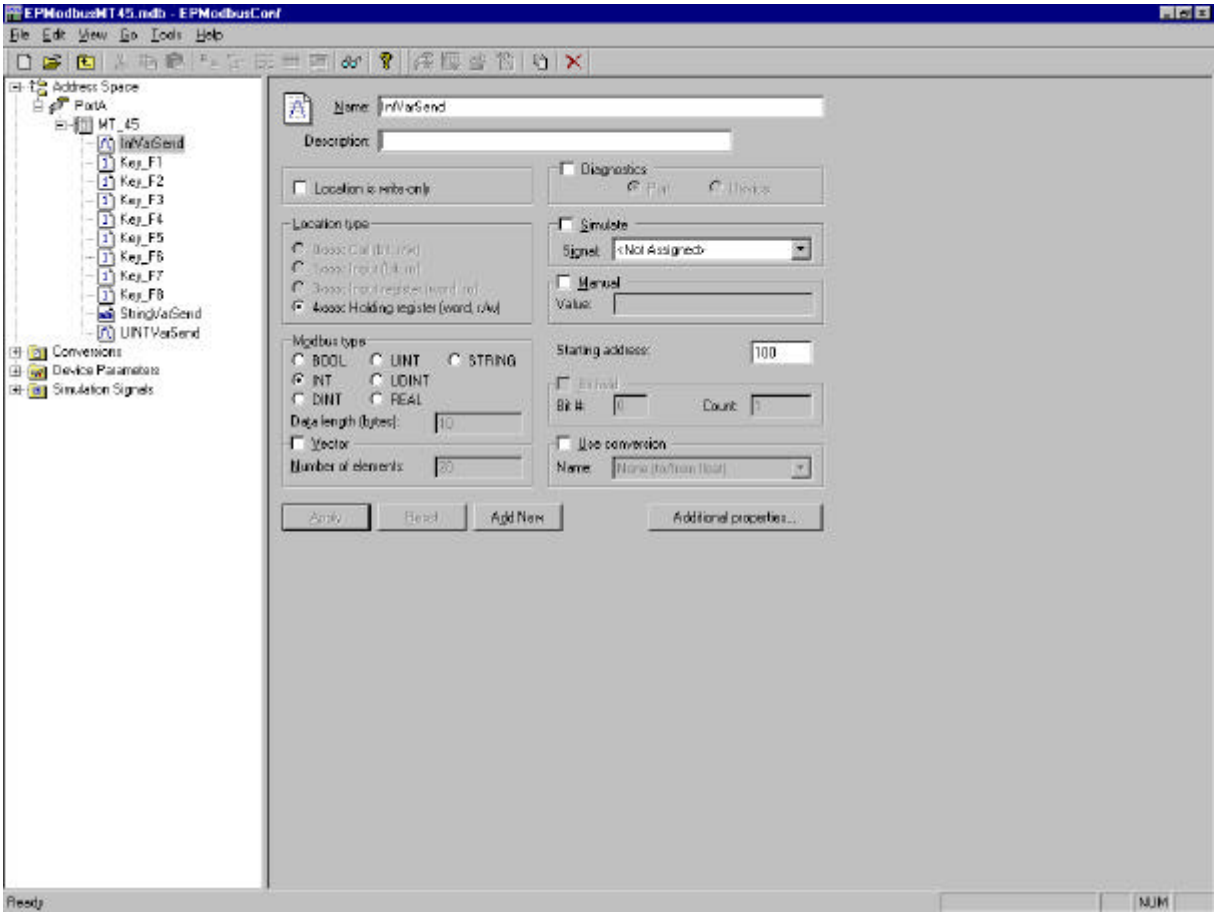


Figure 3 – The EP Modbus Configurator with the parameters of the variable IntVarSend

In the leaves **Data Item** you define the variables with the process connections to the terminal communication port of the OPC Server. The starting address and the type of the variables must be equal with the variables defined in the SPSplusWIN project.

For example: The variable IntVarSend is from the type Integer and has the address 100.

4.4. The OPC DA Custom Client

Please open the OPC client with the start menu:
Start➔Program➔SAE-STAHL Exicom-ProVicom OPC-Server➔OPC DA Custom Client

Open the file **EPModbusMT45.opc** with the command **File➔Open**.
Connect the terminal with the PC, the OPC server is running. After starting the terminal you should see a picture like „Figure 1“. When you have a „Bad“ quality, the communication between OPC server and terminal is not correct. Please check the connection to the terminal and check the communication parameters of the OPC server and the terminal.

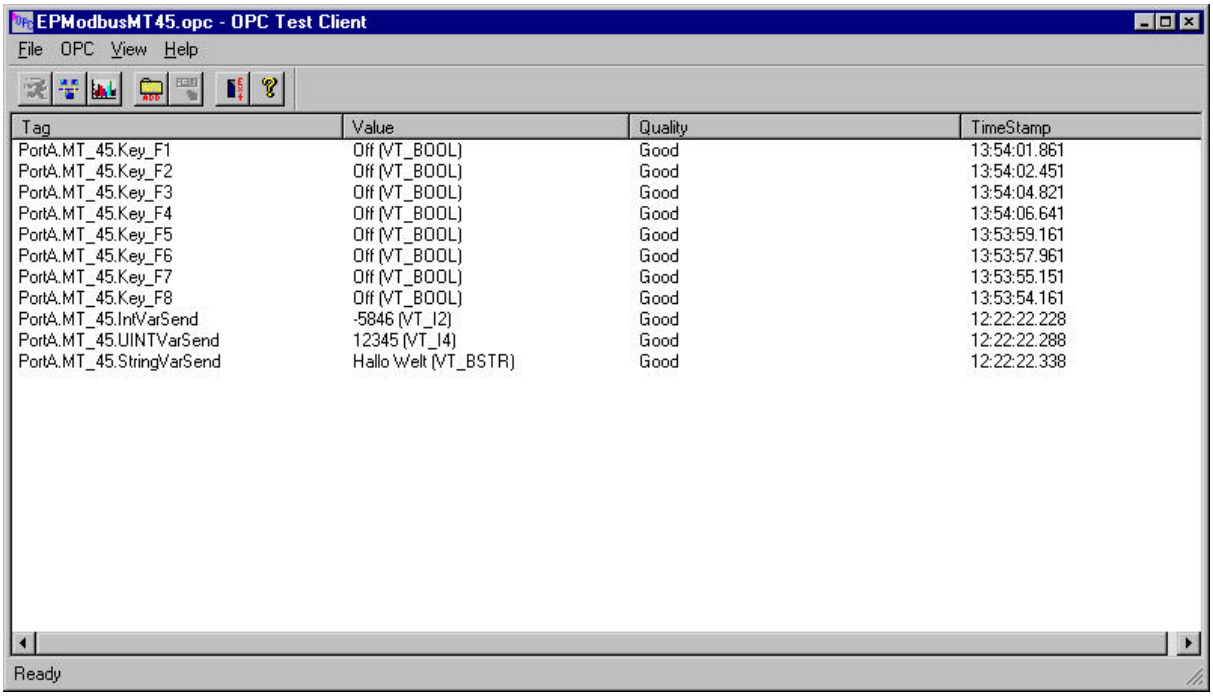

EPModbusMT45.opc - OPC Test Client
File OPC View Help
Tag Value Quality TimeStamp
PortA.MT_45.Key_F1 Off (VT_BOOL) Good 13:54:01.861
PortA.MT_45.Key_F2 Off (VT_BOOL) Good 13:54:02.451
PortA.MT_45.Key_F3 Off (VT_BOOL) Good 13:54:04.821
PortA.MT_45.Key_F4 Off (VT_BOOL) Good 13:54:06.641
PortA.MT_45.Key_F5 Off (VT_BOOL) Good 13:53:59.161
PortA.MT_45.Key_F6 Off (VT_BOOL) Good 13:53:57.961
PortA.MT_45.Key_F7 Off (VT_BOOL) Good 13:53:55.151
PortA.MT_45.Key_F8 Off (VT_BOOL) Good 13:53:54.161
PortA.MT_45.IntVarSend -5846 (VT_I2) Good 12:22:22.228
PortA.MT_45.UINTVarSend 12345 (VT_I4) Good 12:22:22.288
PortA.MT_45.StringVarSend Hallo Welt (VT_BSTR) Good 12:22:22.338
Ready

Figure 1 – The OPC DA Custom Client

You can enter values in the terminal or press the function keys and the changes are displayed in the OPC client. You can enter values in the client with the command **OPC➔Write Value to Item** and the value is displayed in the terminal.

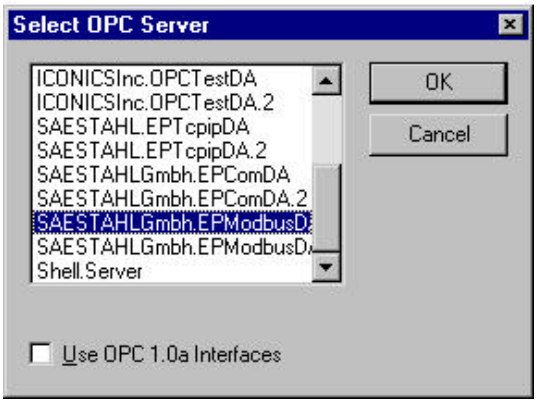


Figure 2 – Connect to the OPC Server

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

When you have made changes in the EPModbus configuration database you can't use the file **EPModbusMT45.opc**. You can open the OPC server with the command **OPC➔Connect**. Please select the OPC server **SAESTAHLGMBH.EPModbusDA**.

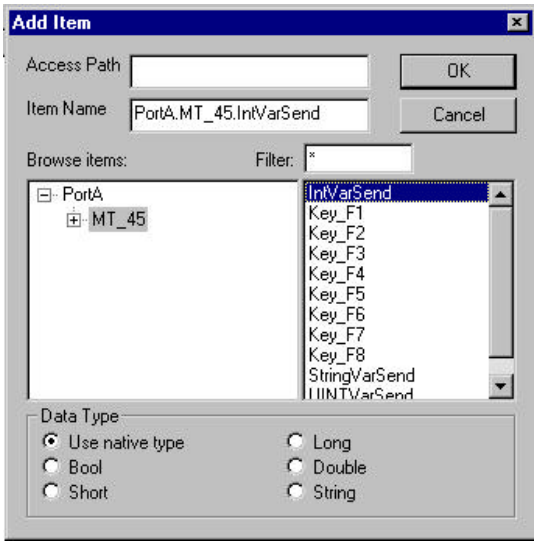


Figure 3 – Add an Item

You can add the data items with the command **OPC➔Add Item**. With this command you can browse through the address space of the OPC server and select the items.

5. Configuration procedure - example application

This chapter describes the step by step procedure of configuring of the Exicom/ProVicom OPC server using configurators. This procedure is illustrated on a simple example.
The configuration procedure has usually 4 steps. They are following:

1. Application specification – specify what the device should do and what the client program should do.

2. Create an application for Exicom/ProVicom devices using SPSPlus Win toolkit (definition of pages, controls, functions, variables, and process connections...).

3. Create a configuration database for EPModbus OPC Server using EPModbus configuration module. For each process connection of the Exicom/ProVicom device define corresponding data item in the server address space.

4. Create a configuration database for EPCom OPC Server using EPCom configuration module. This can be made according to the application purposes. In a very simple application there is created one data item for each data item of the EPModbus with defined Page and Trigger conditions.

5.1. Application specification

MT300 device is used to access a simple MSExcel table using EPCom OPC server. One record of the table has following fields:

Record field	**Data type**
ProductID	*INT*
Name	*STRING*
Quantity	*INT*

MT300 device is used to view and edit records of the database table.

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

5.2. MT300 application project

The application project for MT300 device is placed in the MT300_OPCExample archive file in the SPSPlusWin\Projects directory.
The application has three pages:

- Start page
- ProductID page
- ProductList page

The page *Start* (#0) is displayed after the start of the application. It is used to select one of the following two pages: either the page *ProductID* (#1) or the page *ProductList* (#2).

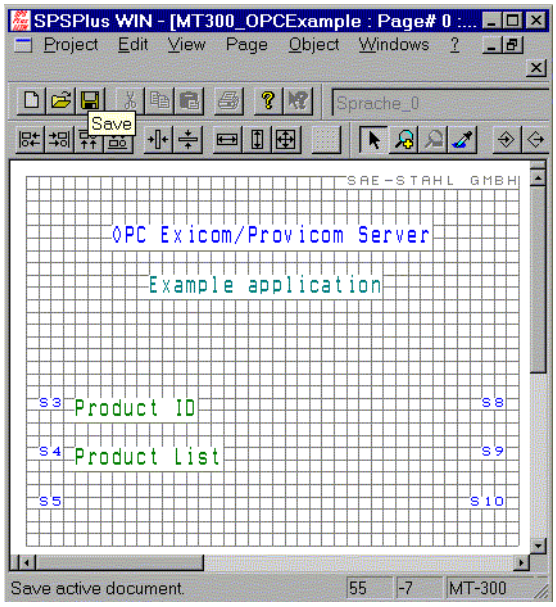


Figure 1 – The 'Start' page

There is on the *ProductID* page one input box to enter the *ProductID* variable and two output boxes to display the corresponding *Name* and *Quantity* variables.

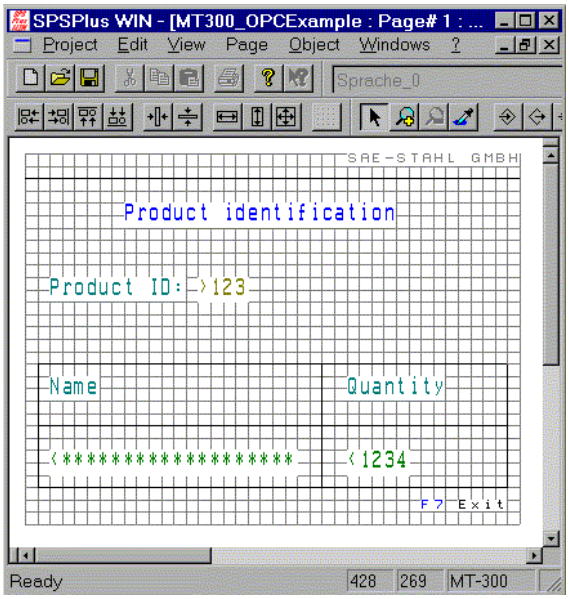


Figure 2 – The 'Product ID' page

If the *ProductID* page is active and the user types a new ProductID to the input box, then the corresponding Name and Quantity should be displayed.

There are on the *ProductList* page 8-output boxes to enter *ProductID_0x* variables (x is a number from 1 to 8), 8 input/output boxes to enter/display *Name_0x* variables and 8 input/output boxes to enter/display *Quantity_0x* variables. Also, there are two output boxes to display the *Count* and *Position* variables. These variables represent the number of all records in database and the current database position.

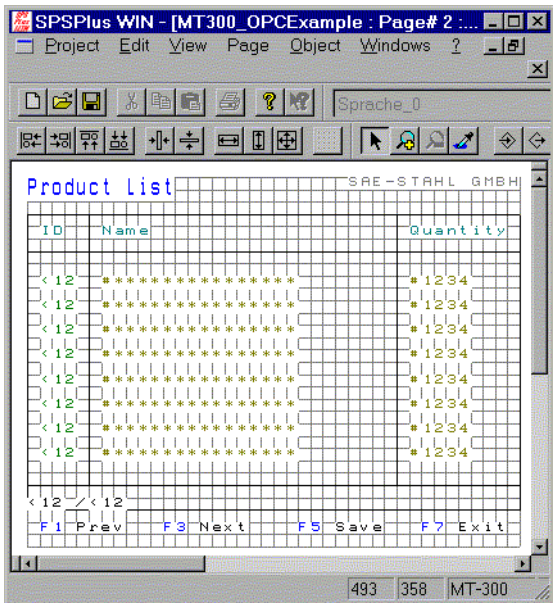


Figure 3 – The 'Product List' page

If the *ProductList* page is active, then the user should use following function keys to view and to edit the table:

Key	***Function***
F1	*Display previous 8 records*
F3	*Display next 8 records*
F5	*Save changed records to the table*

The number of the active page is stored in the *Page* variable. The keyboard status is stored in the *Keyb_SPC_0* variable.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

5.3. EPModbus configuration database

The configuration database of the EPModbus OPC server is stored in the EPModbus_MT300_OPCExample database. It contains corresponding data item for every variable defined for the MT300 application.

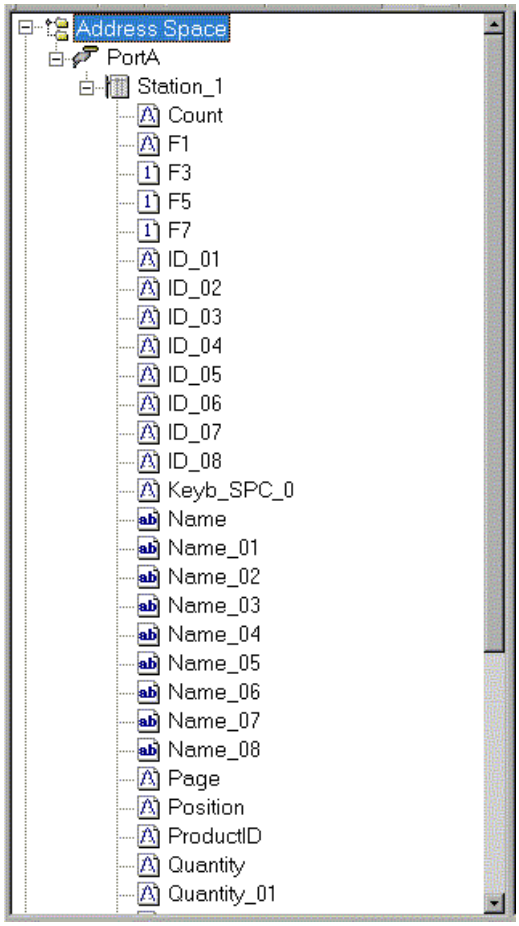


Figure 4 – The address space of the EPModbus OPC server

5.4. EPCom configuration database

5.4.1. Device

The device name is set to Station_1. As a communication driver is selected EPModbus OPC server. Update rate is set to 100 ms.

5.4.2. Data Items

The first data item is *ProductID* that corresponds to the *ProductID* variable of the MT300. The type of the data item is set to *IN* because only the user of the MT300 can change this variable. As a value source the *ProductID* source from the EPModbus OPC server address space is selected. This source should be updated only when the *ProductID* page is active (the page source is equal to 1). The canonical data type of the data item is set to *VT_I2* (integer).

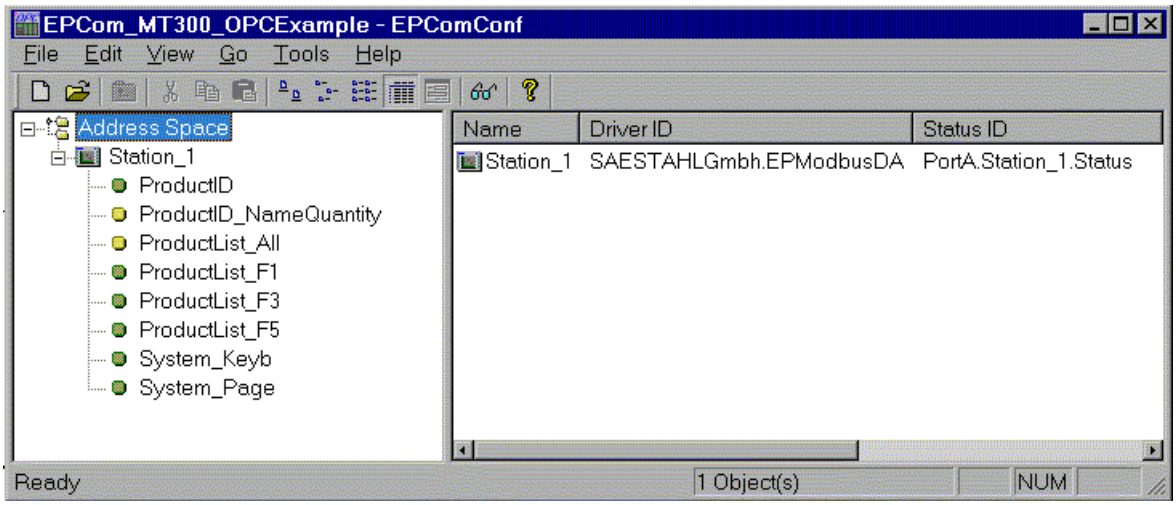


Figure 5 – The address space of the EPCom OPC server

The next data item *ProductID_NameQuantity* corresponds to *Name* and *Quantity* variables on the *ProductID* page. The type of the data item is set to *OUT* because the value of this item should be written by the client program and displayed on the MT300. As value sources *Name* and *Quantity* sources from the EPModbus OPC server address space are selected. These sources should be updated only if the *ProductID* page is active (if page source is equal to 1). The canonical data type of the item is set to *VT_BSTR* (string). The client can write the value for both variables in one string. The values should be delimited with the delimiter character '/'.

The next data item *ProductList_All* corresponds to all variables from the *ProductList* page. The type of the data item is set to *OUT.* It allows access for reading and writing. As value sources all sources from the EPModbus OPC server address space that correspond to the variables placed on the *ProductList* page are selected. These sources should be updated only when the *ProductList* page is active (if page source is equal to 2).

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

The data items *ProductList_F1*, *ProductList_F3* and *ProductList_F5* correspond to the function keys F1, F3 and F5 of the MT300. The type of these data items is set to *IN* – because it has no sense to write to a function key. As value sources corresponding key sources from the EPModbus OPC server address space are selected. These sources should be updated only when *ProductList* page is active (if page source is equal to 2). The maximum of 32 values of the data item should be buffered. The canonical data type of the data items is set to *VT_BOOL*.

The data items *System_Keyb* and *System_Page* correspond to the *KEYB_SPC_0* and *PAGE* variable of the MT300 application. As value sources are selected corresponding sources from the EPModbus OPC server address space. These sources should be updated permanently.

5.5. The OPC client

MSExcel workbook OPCExampleClient is used as a client program. This client was described in the chapter ***0*** *11.2. OPC DA Automation* client**.** The VBA script of this workbook creates a connection with the EPCom OPC server, when the sheet *ProductList* is activated. It creates a new private group in the server and adds following items from the EPCom server address space to the group:

Station_1.System_Keyb
Station_1.System_Page
Station_1.ProductID
Station_1.ProductID_NameQuantity
Station_1.ProductList_All
Station_1.ProductList_F1
Station_1.ProductList_F3
Station_1.ProductList_F5

These items are set to be active. The update rate of the group is set to 1000 ms.

The rest of the time the client processes the subscription callbacks received from the EPCom OPC server.

If it receives the value of the *Station_1.ProductID* item, then it searches for corresponding record in the table and writes the name and quantity to the item S*tation_1.ProductID_NameQuantity*.

If it receives the *Station_1.ProductList_F1* or *Station_1.ProductList_F3* and the values of these items are TRUE, then it prepares a set of table records and writes it to the item *Station_1.ProductList_All*.

If it receives the *Station_1.ProductList_F5* and the value is TRUE, then it reads the current value of the *Station_1.ProductList_All* and makes corresponding changes in the table.

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

6. Introduction to OPC servers

6.1. OLE for Process Control (OPC)

6.1.1. OPC Fundamentals

OLE for process control (OPC) is a standard mechanism for communicating to numerous data sources. It is an open and effective communication architecture concentrating on data access.

OPC is based on Microsoft's OLE/COM (DCOM) technology for data exchange between applications. The OPC was designed by OPC Foundation organization.

The architecture of OPC is a client-server model. OPC servers and OPC clients may be provided by different vendors (vendor independence). More than one OPC clients can be connected to one OPC server at the same time (multi-client access). An OPC Client can connect to OPC servers running on different nodes of the network (network access).

6.1.2. OPC Servers

OPC servers can be divided into three types:

- OPC DataAccess Server
- OPC Alarm&Event Server
- OPC HistoricalData Server

OPC DataAccess Servers provide the access mechanism for OPC Clients to get the current state of the device sources.

OPC Alarm&Event Servers provide the mechanisms for OPC Clients to be notified of the occurrence of specified events and alarm conditions.

OPC HistoricalData Servers provide data compression as well as raw data storage.

6.1.3. OPC Server Interfaces

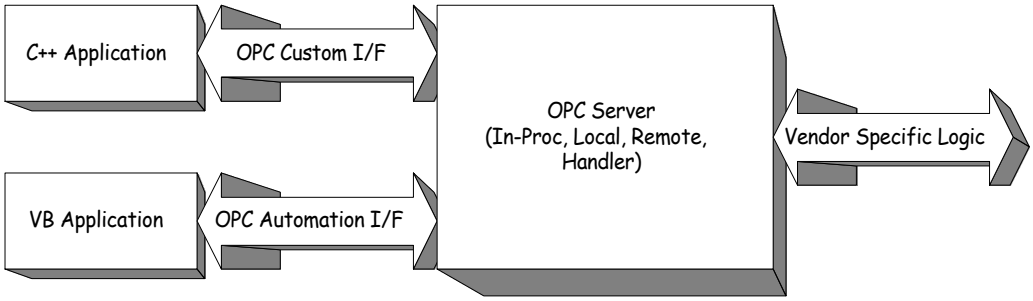


Figure 6 - The OPC Interfaces

An OPC client application communicates to an OPC server through the specified custom and automation interfaces. The custom interface can be used by function pointer oriented languages, such as C++. The automation interface can be used by script languages, such as Visual Basic. OPC servers must implement the custom interface, and optionally may implement the automation interface. OPC Foundation provides a standard automation interface wrapper, which can be used with any custom-server.

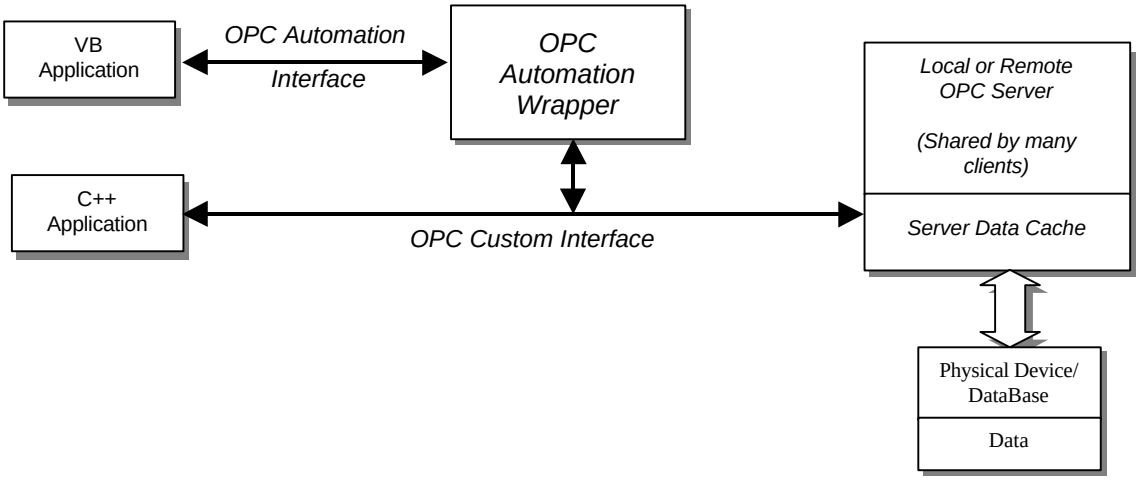


Figure 7 – The OPC Automation Wrapper DLL

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

6.1.4. OPC DataAccess Server

All OPC DA Servers are comprised of several objects: the server, the group, and the item.

The OPC server object maintains information about the server and serves as a container for OPC group objects. It provides functions such as getting of the server status information and browsing of the servers address space. The server address space is a set of all available data items of the server.

The OPC group object enables to group data items. Also, it enables data exchange between OPC server and OPC client. The OPC client can periodically read and write data to the items in the group. Also, the OPC group object enables to create exception based connections between the client and the items in the group. An OPC client can configure the rate that an OPC server should provide the data changes to the OPC client.

The OPC item objects represent connections to data sources within the server. An OPC item, from the custom interface perspective, is not accessible as an object by an OPC client. Therefore, there is no external interface defined for an OPC item. All access to OPC items is via an OPC group object that “contains” the OPC item, or simply where the OPC item is defined.

The OPC item object is accessible as an object only from the automation interface perspective. It enables to maintain the item status and to read and write to the item.

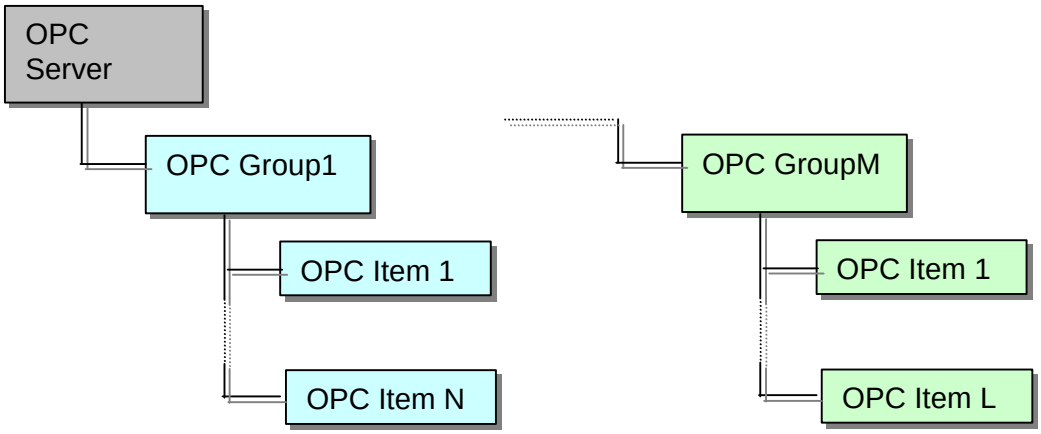


Figure 8 – The hierarchy of the OPC Server objects

All objects – server object, group objects and item objects are COM-objects. The methods of the objects are organized into logical groups – interfaces. These interfaces are common for all OPC servers. The OPC servers must have implemented all required interfaces. Some of the interfaces are optional. Optional interfaces do not have to be implemented. Custom interfaces of the objects are illustrated on the following figure. Interfaces marked in brackets are optional.

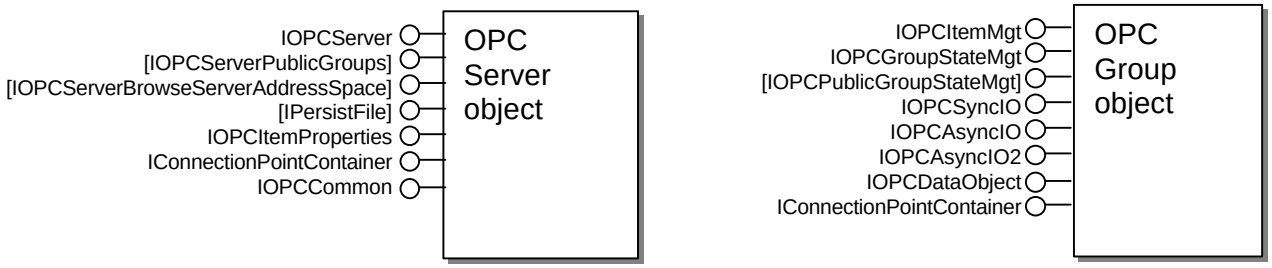


Figure 9 – The interfaces of the OPC server objects

6.2. References

- Microsoft COM specification (available at Microsoft‘s ftp site)
- The OPC Data Access Custom Specification 2.03, OPC Foundation July 1999
- The OPC Data Access Automation Specification 2.02, OPC Foundation February 1999
- The OPC Alarm and Event Access Specification 1.0, OPC Foundation
- The OPC Historical Data Access Specification 1.0, OPC Foundation

7. Exicom/ProVicom OPC Server

7.1. Exicom/ProVicom OPC server

Exicom/ProVicom OPC server is an out-process OPC DataAccess server. It enables OPC clients to communicate with SAE-STAHL Exicom/ProVicom devices.

Exicom/ProVicom devices are operator terminals. Operator terminal is a combination of display and keyboard and it permits man – machine communication.

The server enables an OPC client to send requested data to the display of the terminal and to get data from the user that were entered using terminal keyboard, barcode scanner or serial link.

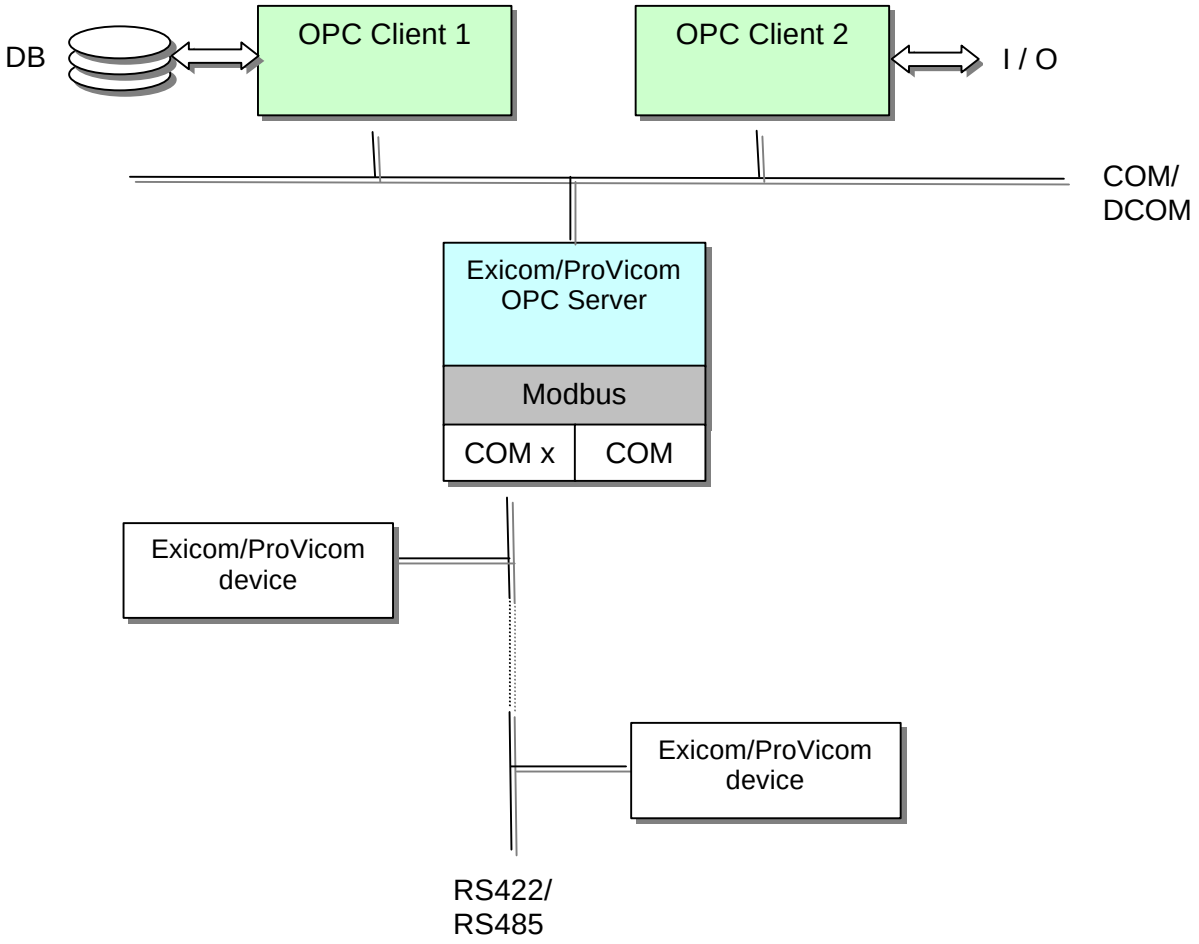


Figure 10 – The Exicom/ProVicom OPC server

The communication between the client and server is based on the COM / DCOM technology. DCOM enables remote connection client-server (over the network).

The server enables to access at the same time multiple Exicom/ProVicom devices through PC serial communication channels. It doesn‘t require any complement hardware. The communication with the devices is controlled by the server. It uses MODBUS communication protocol. The server is working as master and devices behave as slaves. The address space of the server (the device network structure, parameters and the available data items) is fully configurable.

7.2. Implemented OPC interfaces

The Exicom/ProVicom OPC Server implements following interfaces:

Server object	Interface	Version (1.0 / 2.0)	Supported
OPC Server	IOPCServer	1.0 / 2.0	Yes
	IOPCServerPublicGroups	1.0 optional / 2.0 optional	No
	IOPCBrowseServerAddress Space	1.0 optional /2.0 optional	Yes
	IOPCItemProperties	2.0	Yes
	IConnectionPointContainer	2.0	Yes
	IOPCCommon	2.0	Yes
	IPersistFile	1.0 optional / 2.0 optional	Yes
OPC Group	IOPCGroupStateMgt	1.0 / 2.0	Yes
	IOPCPublicGroupStateMgt	1.0 optional / 2.0 optional	No
	IOPCASyncIO2	2.0	Yes
	IOPCAsyncIO	1.0	Yes
	IOPCItemMgt	1.0 / 2.0	Yes
	IConnectionPointContainer	2.0	Yes
	IOPCSyncIO	1.0 / 2.0	Yes
	IDataObject	1.0	Yes

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

7.3. Exicom/ProVicom OPC server architecture

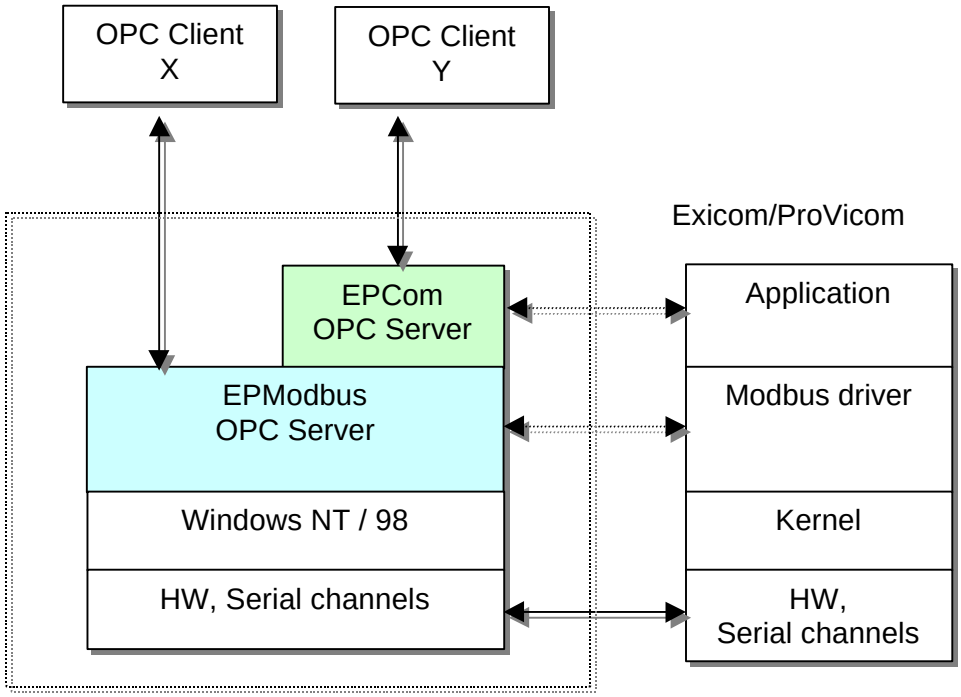


Figure 11 – The architecture of the Exicom/ProVicom OPC server

The Exicom/ProVicom OPC server consists of two levels:

- EPModbus OPC Server
- EPCom OPC Server

EPModbus OPC server implements MODBUS communication protocol. This server enables to access each variable defined in the application of the Exicom/ProVicom device. It allows the communication simultaneously on 4 communication channels. Up to 247 devices can be connected to each channel.

EPCom OPC server controls the underlying EPModbus OPC server according to the user-defined conditions. For example it enables to transfer data only from variables defined on the current page of the device or only if a key was pressed and so on. This can minimize the transfer on the communication line to necessary activities. Also EPCom server enables to create records of more variables. This on the other hand can minimize the transfer between the client program and the server.

The client program should use only one of these servers: either EPModbus or EPCom. EPCom itself is an extension of the EPModbus server and implements some useful functions. If the client uses the EPCom server, the EPModbus server is used automatically by the EPCom server. EPCom server is an OPC client of the EPModbus server.

7.4. Address space and configuration database

Address space of an OPC server is a set of data items that any OPC client program can access. The address space of the Exicom/ProVicom OPC server can differ according to the application of the connected Exicom/ProVicom devices. Also, it can differ according to the number of devices, the number of used communication channels and so on.

The address space is defined in a configuration database. This database is loaded at the startup of the server. Both servers (EPCom and EPModbus) have their own configuration database.

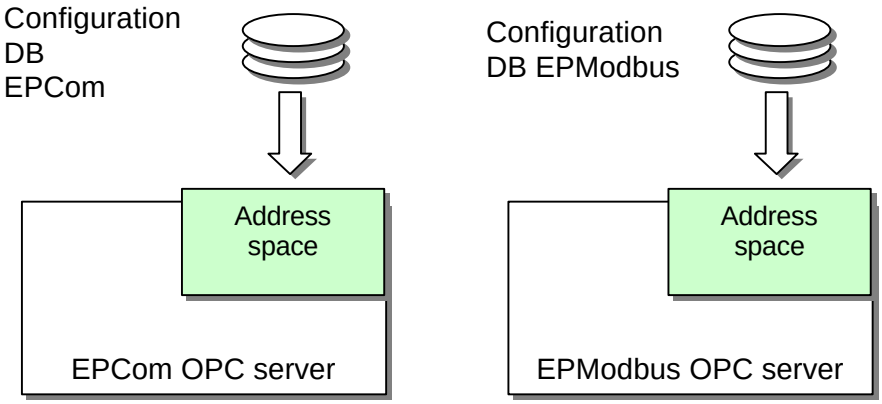


Figure 12 – The address space and configuration database

The applications of Exicom/ProVicom devices can be created using SPSPlus Win. Also, SPSPlus Win can generate from application projects the configuration databases of the OPC Server.

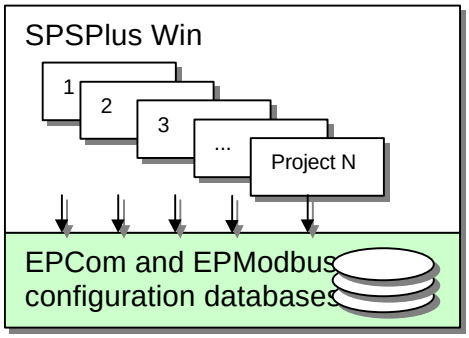


Figure 13 – The generating of the configuration databases

Each application project defines a set of visualization objects (pages, input-output boxes, lists, and buttons...) and corresponding process connections (memory type, memory address, data type, and length in bytes...). Information from the application projects is used to create configuration database for the OPC server. The OPC server with the generated configuration database is able to communicate with the corresponding Exicom/ProVicom devices.

The configuration databases can also be created manually. There are configuration modules *EPModbus configurator* and *EPCom configurator* that can be used to create and to modify the configuration databases.

For both EPModbus and EPCom OPC servers can be created more then one configuration databases. But only one of them can be active. To select the active configuration database from existing databases can be used *EPModbus administrator* and *EPCom administrator*.

7.5. EPModbus OPC server

7.5.1. Features

EPModbus OPC server implements modbus communication protocol.

Basic features of the server are following:

- *support of both RTU and ASCII modbus modes*
- *4 serial channels*
- *247 devices per channel*
- *data transfer optimizations*
- *simulation*
- *automatic conversions from device scale to engineering units*

7.5.2. Address space hierarchy

The address space tree of the EPModbus OPC server has following hierarchy:

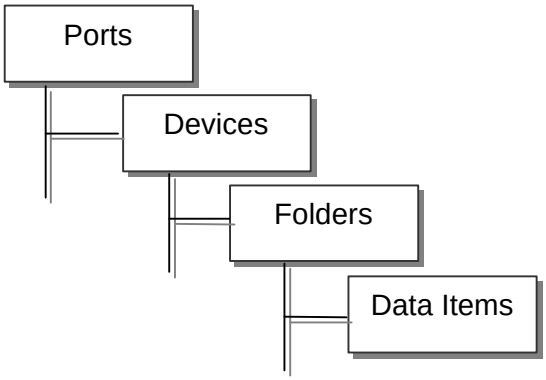


Figure 14 – The hierarchy of the address space

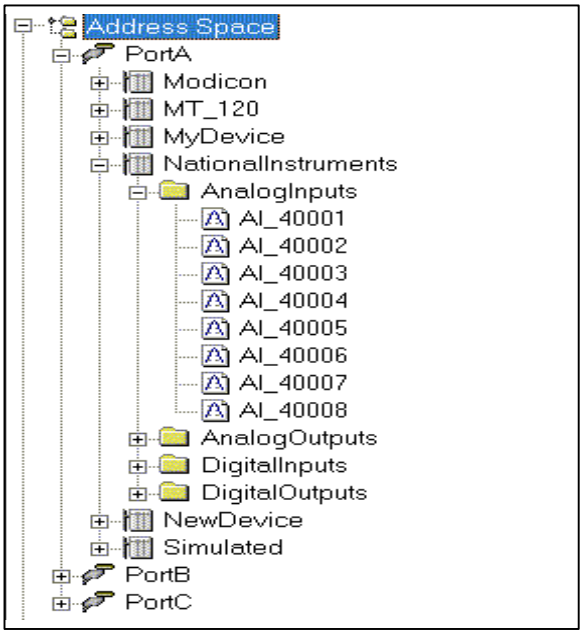


Figure 15 – The address space of the EPModbus OPC server - example

Port is equivalent to a physical serial port in the computer.

Device is equivalent to a physical device connected to the serial port.

Folder is an object that can group items that logically belong together. It does not have any direct representation in the Modbus protocol. Folders can be used any way that is suitable to the application (for example folder of analogs, digitals and so on.). Three levels of folders are allowed.

Data Item represents a register in the device or a range of registers.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

7.5.3. How the server works

An OPC client can connect to the server, browse the address space and logically organize requested items to groups. The client can create any organization of items. The way that the items are organized can influence the performance of the server.

The server periodically updates each group of items. The group parameter *RequestedUpdateRate* specifies the update period. This parameter is common for all items in a group and is set by the client program.

The server uses optimization methods while reading values from the devices. The server tries to optimize the communication with the devices by requesting as much data as possible in one message. This optimization can be used only for one device at the same time. For this reason it is recommended to organize items of the same device in the same group.

If the server detects that a device doesn‘t response, then the device is suspended and the corresponding items are not updated. The server periodically tries to connect to the devices at specified ‚*suspend period*‘. This approach minimizes the update cycle while some of the devices are disconnected.

7.6. EPCom OPC server

7.6.1. Purpose

Exicom/ProVicom devices are operator terminals – they have a display and a keyboard. The application of the terminal usually uses hundreds of variables that are organized in many pages. The user goes through pages using keyboard, displays current values and types new values of the variables. The task of the OPC server is to cache the current state of the variables. The server can update variables in two different ways:

1. It periodically reads the current state of all variables.

2. It periodically reads the current state only those variables that can be changed by the user (input variables displayed on the current page, keyboard state, system variables required to be updated permanently and so on).

The first way is simple but it is convenient only for small amount of variables. The second way is optimal for rising number of variables and devices.

EPCom OPC Server implements some useful functions that solve this problem. EPCom OPC server is an extension of the EPModbus OPC server. If the client uses the EPCom server, the EPModbus is used automatically as the communication driver (but this can be changed in the configuration database). EPCom server controls the underlying EPModbus according to **user-defined conditions**. It enables for example to transfer data only from variables defined on the current page of the device or if a key is pressed. This can minimize the transfer on the communication line only to necessary activities. EPCom enables also to group more sources from EPModbus into one data item (e.g. data item can represents all variables of one page).

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

7.6.2. Address space hierarchy

The address space tree of the EPCom OPC server has following hierarchy

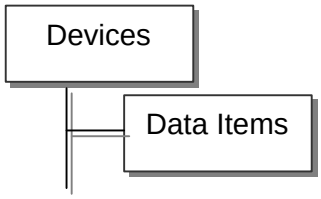


Figure 16 – The hierarchy of the address space

Device corresponds to one Exicom/ProVicom device.

Data Item groups one or more OPC sources into one item. It corresponds to one or more Exicom/ProVicom variables (e.g. placed on the same page).

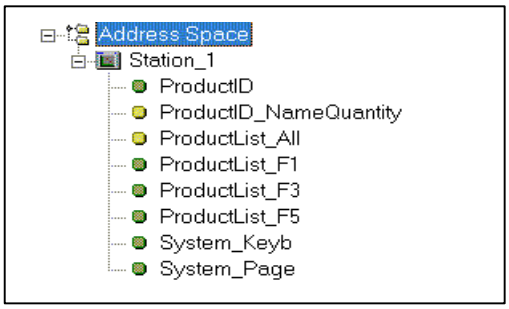


Figure 17 – The address space of the EPCom OPC server - example

8. Exicom/ProVicom OPC server configurators

8.1. User interface

User interface of the configurators is Microsoft Explorer-like, and it provides numerous functions to make the configuration easier, such as Copy and Paste, Drag and Drop, or context based popup menus.

- Views layout
- File menu
- Edit menu
- View menu
- Go menu
- Tools menu
- Popup menus

These functions are the same for both EPModbus and EPCom configurators.

8.1.1. Views layout

Views layout is divided into four view panes:

- Tree view
- List view
- Dialog-like view
- OPC Data Access Monitor view

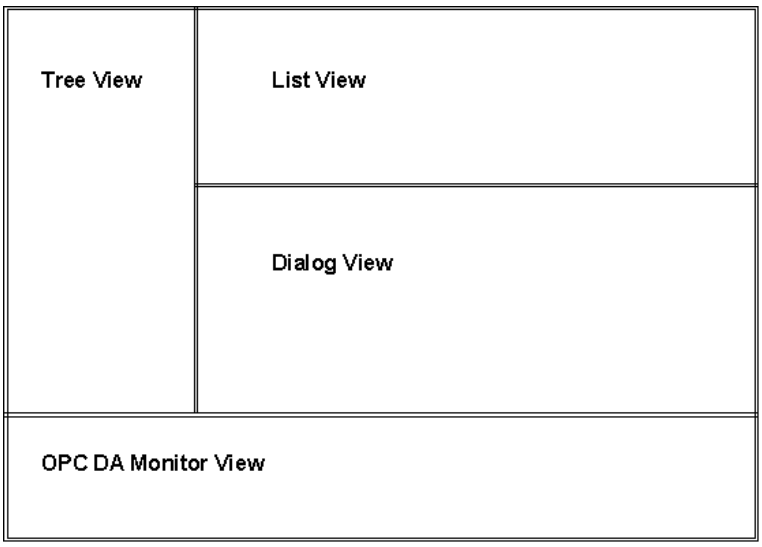


Figure 18 – The views layout

The tree view is used to explore the structure of the address space, device parameters, conversions and simulation signals.

The list view is used to display the contents of the item selected in the tree view.

The dialog like view is used to edit parameters of the item selected in the tree view.

The monitor view is used to monitor connection with the runtime module and to display current values of the items, selected in the tree view. This is useful for testing purposes.

8.1.2. File menu

File menu provides commands for creating (New), opening (Open…) and saving (Save As…) configuration as Microsoft Access Database file (.mdb). These commands work in a standard way.

The Exit command will close the configurator.

Other file menu commands are following:

- Windows CE Export command
- Export CSV submenu
- Import CSV submenu
- Make Active command

8.1.2.0.1. Windows CE Export command

This command exports the whole configuration into a single text file. Then this file can be transferred to a Windows CE device and used by the server runtime running there. This additional export step is necessary, as Windows CE do not support .mdb or other database formats.

8.1.2.0.2. Export CSV submenu

Commands in this submenu allow to export different parts of the configuration into a standard CSV (comma separated variables) file.

8.1.2.0.3. Import CSV submenu

Commands in this submenu allow to import different parts of the configuration from a standard CSV (comma separated variable) file.

8.1.2.0.4. Make Active command

The active database is the database that the runtime part of the server will use when started. The active database may (but does not have to) be the same as the database currently edited in the configurator. You can work on any database inside the configurator, while the runtime part has its own active database.

The Make Active command sets the currently edited database as the active database. Next time the server runtime part starts, it will use this active database for all its operations. This menu item is disabled when the currently edited database IS already active.

Before the active database is actually set, the program asks you to confirm the setting. The dialog box that is invoked looks like this:

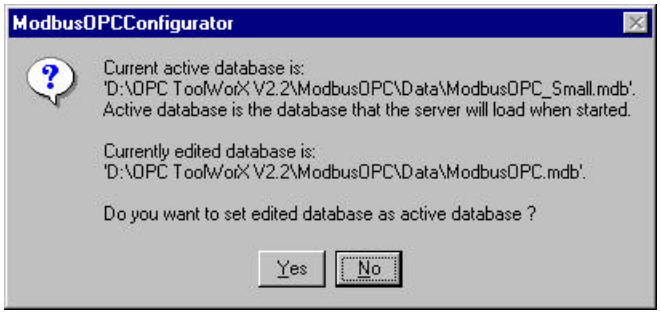

ModbusOPCConfigurator
Current active database is:
'D:\OPC ToolWorX V2.2\ModbusOPC\Data\ModbusOPC_Small.mdb'.
Active database is the database that the server will load when started.
Currently edited database is:
'D:\OPC ToolWorX V2.2\ModbusOPC\Data\ModbusOPC.mdb'.
Do you want to set edited database as active database ?
Yes
No

Figure 19

You can now verify the database file names and decide to proceed or cancel the operation.

8.1.3. Edit menu

The Edit menu contains many standard commands, such as Rename, Delete, Cut, Copy, Paste, Select All and Invert Selection.

The configurator supports standard drag-and-drop behavior in the list and tree views.
Other edit menu items are following:

- New submenu
- Multiply command

8.1.3.0.1. New submenu

This submenu contains commands to add new objects into an active branch in the tree. The actual content of this submenu is based on the type of the branch you have selected.

8.1.3.0.2. Multiply command

Multiply command within Edit menu belongs to the set of non-standard commands. Its purpose is straightforward. Selecting this command will launch Multiply Item Dialog.

The actual look and contents of the dialog is server-dependent. The simplest example of this dialog is on the picture below:

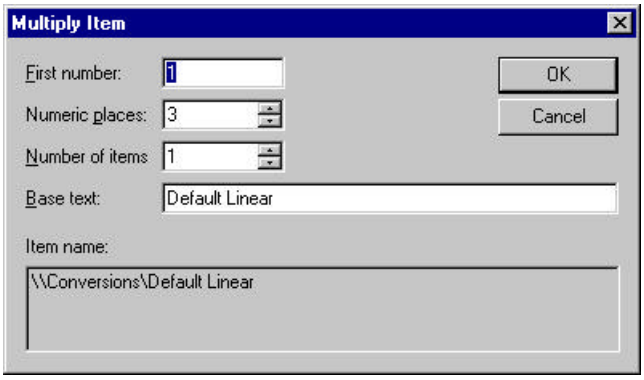


Figure 20 – Multiply item command

First Number	*The number appended to the first generated item name. Following items will be numbered consecutively.*
Numeric places	*The count of digits of the number appended to the item name*
Number of items	*The count of items to be generated*
Base Text	*String that the generated item names will start with*
Item name	*Path to data item in the left tree view pane*

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

8.1.4. View menu

View menu allows users to show/hide toolbars such as Standard Buttons and Data Manipulation Buttons, or a status bar.

The items in the list view pane could be displayed in one out of four following modes: Large Icons, Small Icons, List and Details.

View panes could be shown or hidden selecting the appropriate command from the menu or pressing F11 for Dialog View or F12 for Monitor View pane.

Standard Select Language dialog allows the user to adjust the international environment according to his/her needs.

Choose Show/hide columns to adjust report view in the right pane of the configurator. Checking appropriate columns within Sort can influence the order of data items in the right pane by submenu commands.

8.1.5. Go menu (browsing the data)

To view available records in any selected window, use a command from a list that appears in the Go menu, or use the Arrow, PageUp/Down, Home, End keys for moving in the tree view in combination with Alt key.

Up One Level command moves the cursor one level closer to the root of the tree.

Next Pane or Previous Pane allows users to loop through panes, if more than one is displayed.

8.1.6. Tools menu

8.1.6.0.1. Options command

Options dialog within Tools menu contains Workspace parameters.

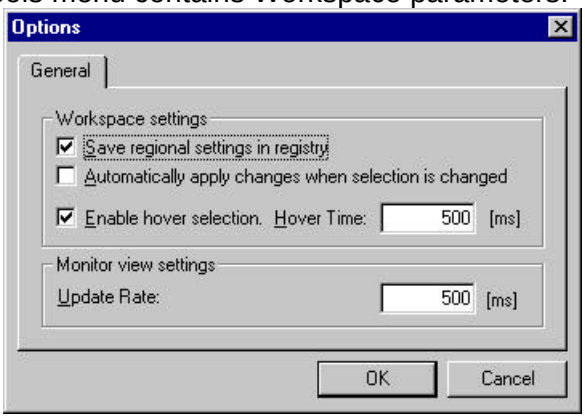


Figure 21 – Options command

Save regional settings in registry	*enables/disables the configurator to save active language chosen from the Select Language dialog into registry and set it up during initialization*
Automatically apply changes when selection is changed	*If checked, the configurator itself tries to update the edited data without requiring clicking on Apply button, and displaying the confirmation dialog.*
Enable hover selection, Hover time	*Allow configurator support automatic selection of the item above which the mouse is hovering. Enter Hover time parameter in milliseconds*
Update rate	*Specifies the update frequency of Monitor View items.*

8.1.7. Popup menus

Instead of using the standard menus to find the command you need, you can also use the right mouse button clicking on the item in the list or tree view.
Popup menu that appears shows the most frequently used commands that are New, Rename, Multiply, Cut/Copy/Paste, and Dialog/Monitor View.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

8.2. Configuration of the EPModbus address space

Modbus is a serial communication protocol allowing connection of 247 devices on a serial line. There is always one device controlling the communication (Master). The rest consists of slave devices (Slaves). Every device is identified by its unique address. The modbus protocol enables to access four register types - *Inputs* (1 bit, read - only), *Input Registers* (16 bits, read-only), *Coils* (1 bit, read-write) and *Holding Register*s (16 bits, read-write). Each register type has 16-bit address space.

The address space tree of the EPModbus server has following levels:

- Ports
- Devices
- Folders
- Data items

8.2.1. Ports

Port is equivalent to a physical serial port in the computer. To correctly configure port means to set *file name* (port), *communication speed* (baudrate), and protocol characteristics such as *transmission mode*, *RTS flow control*, *Stop bits* and *Parity scheme*.
If *Parity checking enabled* is unchecked, then the server ignores the parity bits in the message.

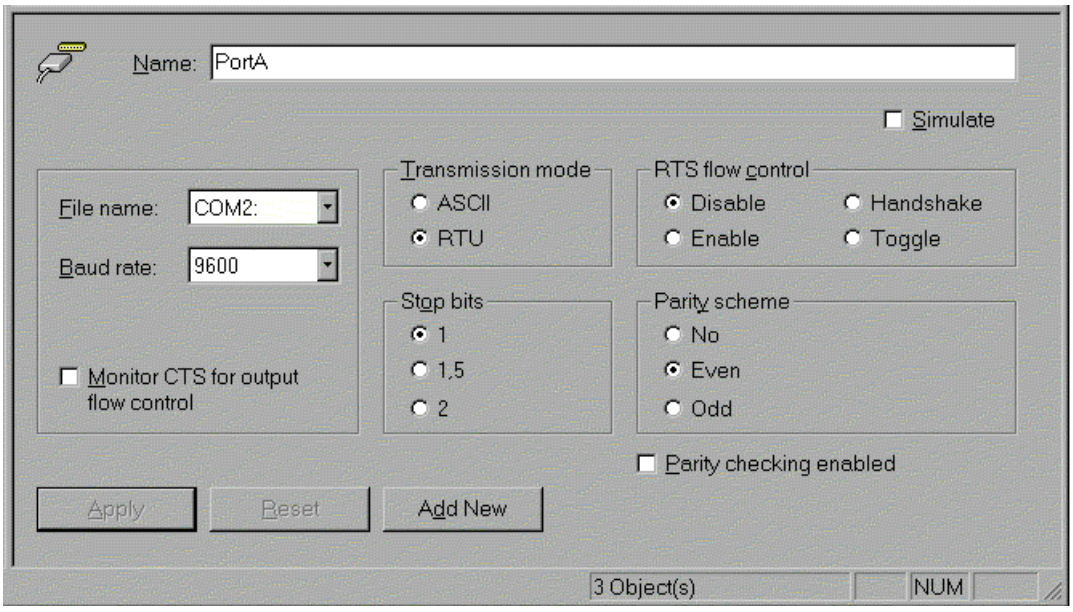


Figure 22 – The ports configuration table

8.2.2. Devices

Every device is connected to particular port. It logically creates second level in the address space tree. The device is represented by its symbolic *name*. Also, it is uniquely identified by the *address*. It is impossible to have two devices with the same address connected to one port.

Setting up Device requires to configure:

- unique address
- device type
- timeouts
- optimization parameters.

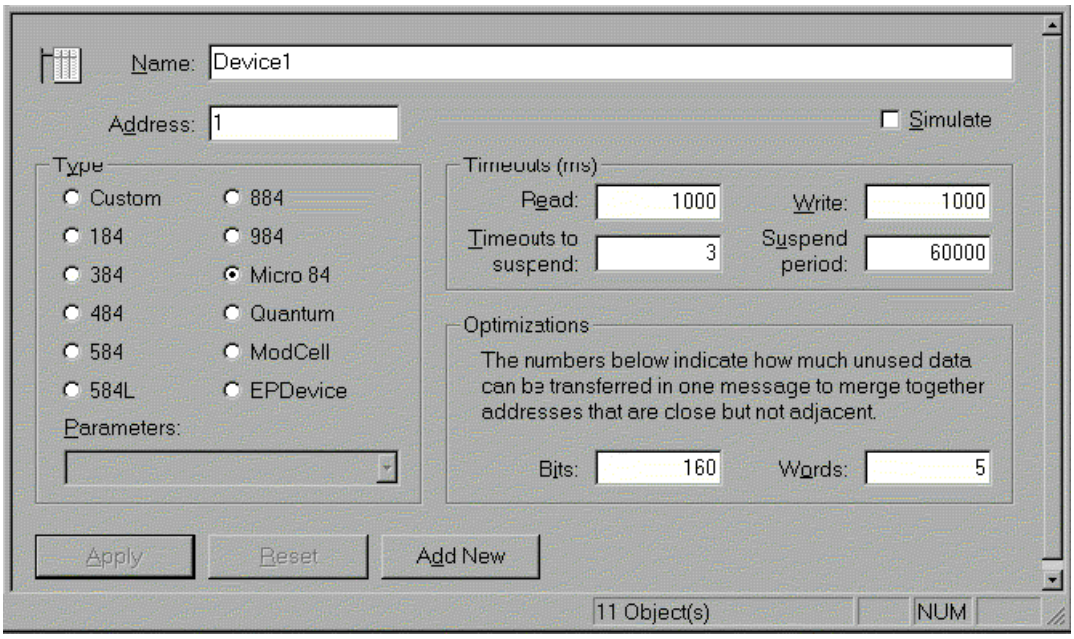


Figure 23 – The 'Devices' configuration table

There is a group of six predefined *Modicon* device types enhanced with *EPDevice* and *Custom* options. *EPDevice* is a short name for Exicom/ProVicom SAE-STAHL device type.

If you have devices that are not among the list of pre-defined types, use *Custom* option and select from *Parameters* combo one of the user-defined device types. For instruction on how to create new or edit user-defined device see also device parameters.

Timeout parameters (separately for read and written messages) specify the period length that server waits for response from devices.

The server tries to optimize the communication with the devices by requesting as much data as possible in one message. Consecutive registers are merged together into one request for efficiency. The server also can read registers that are not really requested, if this allows it to join two blocks of requested registers. The numbers entered under Optimizations specify the maximum block length of adjacent unused data.

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

8.2.3. Folders

Folder is an object that can group items that logically belong together. It does not have any direct representation in the Modbus protocol; you can design and use folders any way that is suitable to your application. Three levels of folders are supported in the EPModbus OPC Server.

8.2.4. Data Items

Data Item represents a register in the device or a range of registers. A symbolic name and description is associated with the data item. OPC Client can obtain the data item description.

New data item creation requires configuration of the following properties:

- Location type
- Modbus type
- Starting address
- Use conversion
- Simulation
- Manual

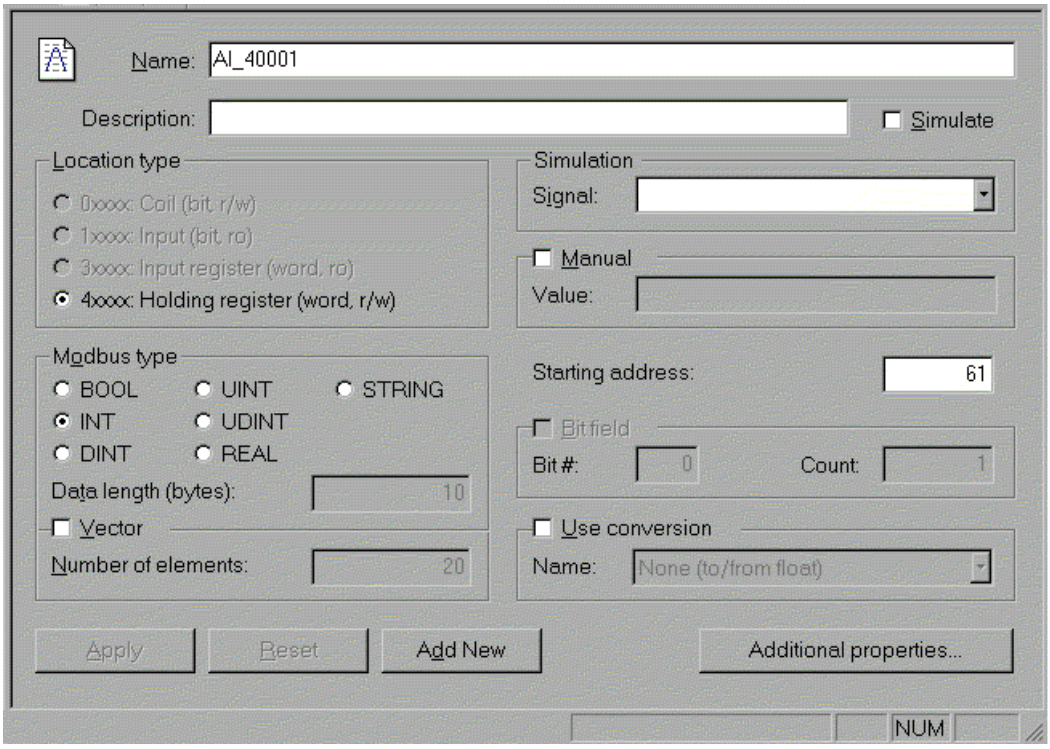


Figure 24 – The 'Data Items' configuration table

Location type

Location type is a type of a register in the device. Device registers are divided into Coils, Inputs, Input Registers and Holding Registers. This table explains the name conventions used:

	Read Only	**Read / Write**
1 bit	*Input*	*Coil*
16 bit	*Input Register*	*Holding Register*

8.2.4.0.1. Modbus type

The location type (device data) will be understood as Modbus type (OPC data type). Modbus data type also depends on the Location type selected.

Example: Coil or Input (1 bit) device data type could be Modbus BOOL only.

When selecting Modbus String type, user has to specify the data length (how many bytes will the String be represented by).

8.2.4.0.2. Starting address

This value specifies the data item address (register number) in the device data space.

With the UINT Modbus type, it is possible to extract bits from the register and use them as a Boolean or integer value. The user can specify a group of *Count* adjacent bits inside a word starting with *Bit #*. This way, it is possible to use a register for several separate data items.

Addresses 65533, 65534 and 65535 of the device data space are reserved for OPC server. OPC server uses registers 65533 and 65534 to store diagnostic information.

- Register 65533 stores the current state of the device
  0 (Device is OK), 1 (Device is suspended)
- Register 65534 stores the current state of the port
  0 (Port is OK), 1 (Port is not available)

8.2.4.0.3. Use conversion

To get the data value converted according to a form prescribed, choose one of the predefined or user-defined conversions.

8.2.4.0.4. Simulation

To test the client functionality, choose a *Simulation Signal* from the Signal drop-down and check *Simulate* check box.

All levels in the Address space (port, device, folder, data item) support process of simulation (*Simulate* check box). Parent list in the tree is superior, has higher priority when deciding to simulate the data item or not. In other words, data item is simulated, if it itself has a Simulation checked, or if any of its parents has *Simulate* checked (it may be simulated even if its *Simulate* check box stays unchecked).

8.2.4.0.5. Manual

If checked, data item will offer constant parameter value, because *Manual* setting is of the highest priority.

The changes in the configuration take effect only when the server reloads the configuration (on startup).

8.2.5. Device Parameters

Device Parameters directory contains the list of custom device types. Device parameters influence the behavior and performance of the server for the device.

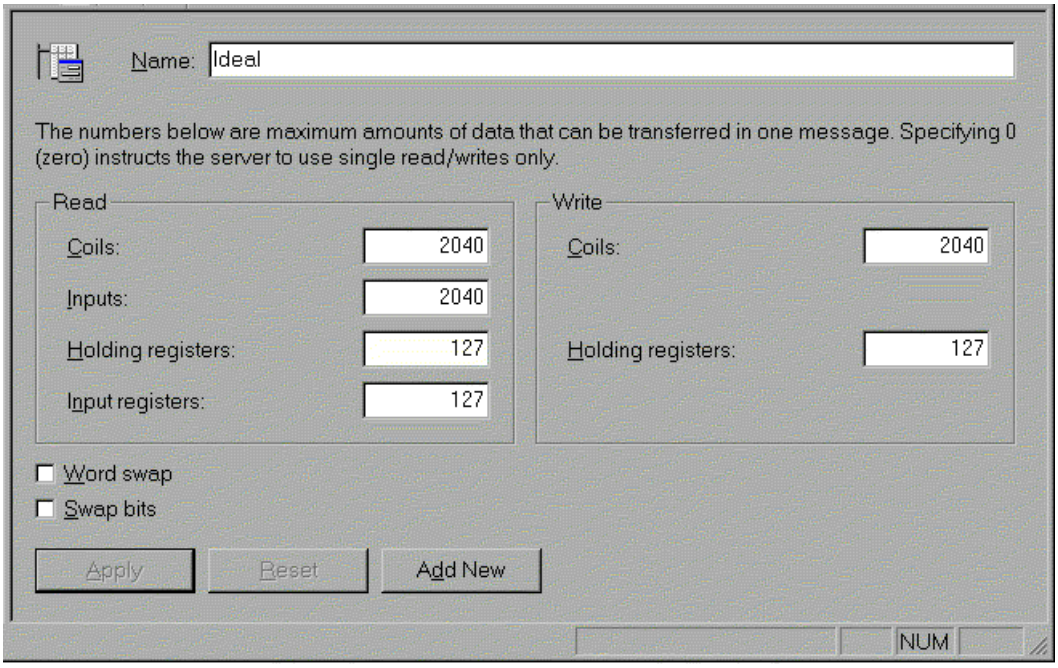


Figure 25 – The 'Device parameters' table

The meaning of numbers in the device parameters dialog is the maximum amount of data that can be transferred in one message. Setting the value equal to zero means force the server to use single read/write messages only.

The server also allows configuring following additional properties for each device type:

Word swap	*Swaps the first word with the second when reading/writing DINT, UDINT or float values.*
Swap bits	*Reverses the order of bits in word-sized values (least significant bit becomes the most significant bit).*

8.2.6. Conversions

Using Conversions directory you can define your own conversions for converting of device data values from instrument range units (IR) to engineering units (EU).

To define a new conversion form you have to specify following properties:

Type of conversion	*No conversion converts the data into float data type, but does not change the value itself.* *Linear or square root conversions keep a linear or square root relation between EU and IR.*
Conversion parameters	*Note that definition of range limits helps some client applications (e.g. when creating slider) and makes sense even when no conversion is specified.*
Clamping	*If clamping is on, the data value will be limited to its High clamp/EU value, when it exceeds the upper limit, and similarly with Low clamp parameter.*

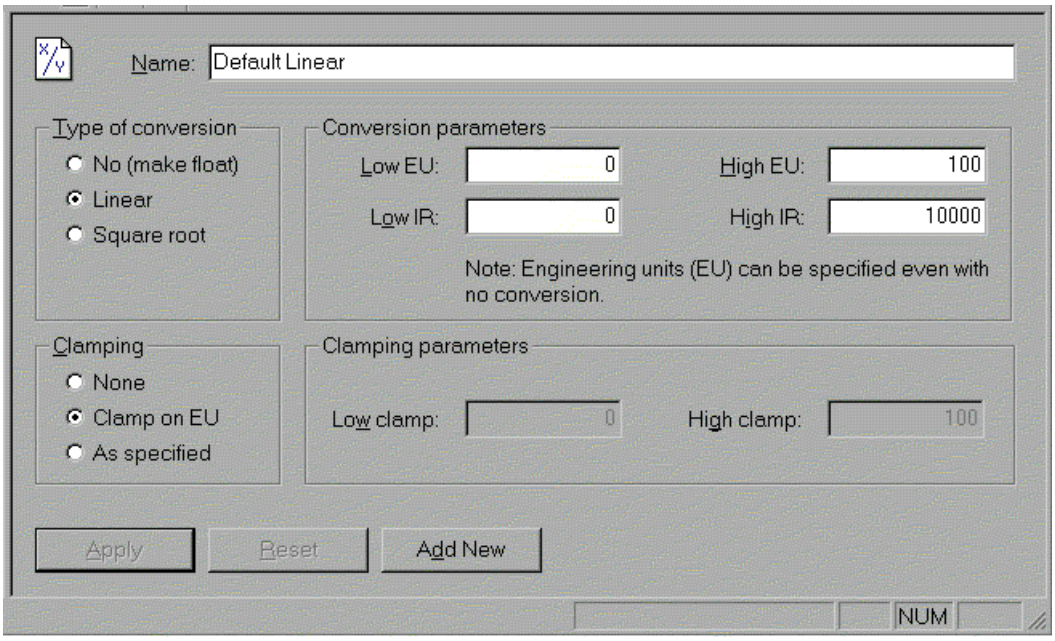


Figure 26 – The 'Conversions' table

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

8.2.7. Simulation Signals

Using simulation signals directory, you can define your own simulation signals. The new signal can be created from predefined set of signals by changing of some of their parameters.

The predefined types of simulation signals are following:

Read count	*is incremented by one every time when the item is read*
Write Count	*increments when the item is written*
Random	*generates random value within the **Amplitude** range starting with **Position***
Ramp, Sine	*are periodical signals. Their time behavior is influenced by **Period** and **Phase** parameters. **Period** specifies the signal frequency, while Phase moves the signal origin on the time axis.*
Square, Triangle	*are periodical signals. Their time behavior is influenced by **Period**, **Phase** and **Ratio** parameters. Ratio defines Triangle signal steepness, or Square signal H/L proportions*
Step	*is a periodical signal. Their time behavior is influenced by **Period**, **Phase** and **# of steps** parameters. **# of steps** parameter defines a number of steps the signal amplitude will be divided into.*

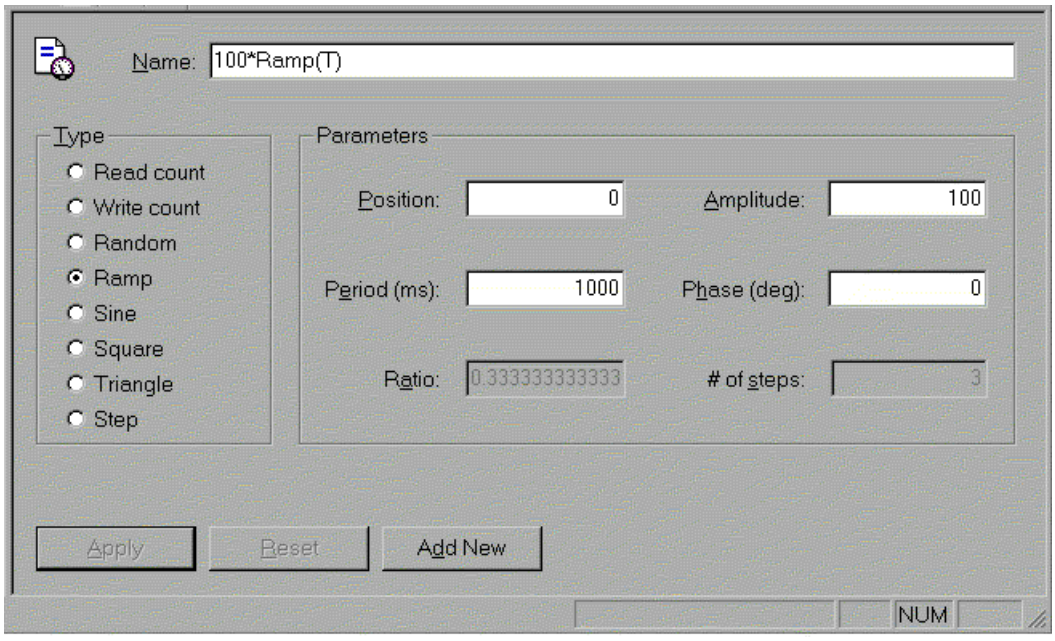


Figure 27 – The 'Simulation signals' table

8.3. Configuration of the EPCom address space

The address space tree of the EPCom server consists of two levels – Devices and DataItems. *Device* represents a branch and corresponds to one Exicom/ProVicom device (EPDevice). *DataItem* represents a leaf and corresponds to one EPDevice source (e.g. variable defined in EPDevice). It can correspond also to a group of EPDevice sources.

The address space should be defined according to the connected EPDevices and their application purposes.

8.3.1. Device

Setting up *Device* requires to configure following parameters:

- Driver ID
- Update rate
- Deadband
- Status item ID
- Update status period

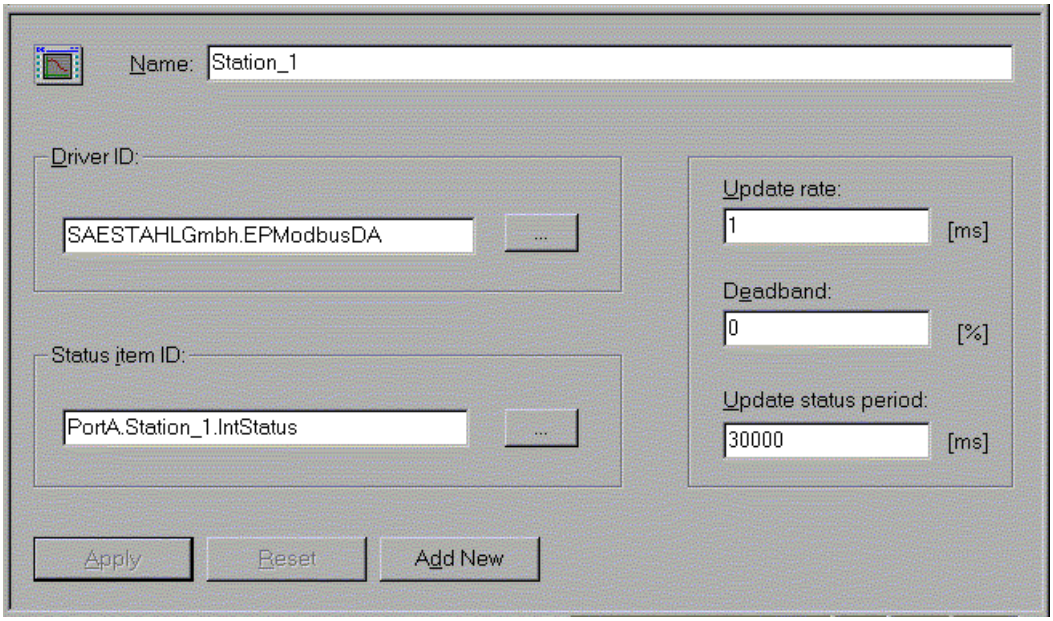


Figure 28 – The 'Device' configuration table

8.3.1.0.1. Driver ID

Driver ID defines a communication driver that will be used to read data from and write data to EPDevice sources. The communication driver can be any OPC DA server (2.0). This driver should be selected according to the communication protocol of the corresponding EPDevice. The protocol of the EPDevice is specified during a design of the EPDevice application.
It is recommended to use EPModbus OPC Server for the modbus communication protocol.

8.3.1.0.2. Update rate

This parameter defines the requested rate of updating of the EPDevice sources.

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

8.3.1.0.3. Deadband

EPCom server is connected to the communication driver using *subscription callbacks*. The communication driver automatically updates selected sources in its internal cache. The parameter *Deadband* informs the driver about the percent change in a source value that will cause a subscription callback for that value to the EPCom server. This parameter only applies for ‚Analog' sources. If deadband is 0.0, then all changes are sent to the EPCom server.

8.3.1.0.4. Status item ID

If specified, then EPCom server periodically writes its current status to this source.
Status information values:

0 - reserved
1 – server running
2 – reserved
3 – reserved

8.3.1.0.5. Update status period

Specifies period of updating of the status source defined in the Status item ID field.

8.3.2. Data Items

DataItem corresponds to one EPDevice source (e.g. variable in the EPDevice). It can also correspond to a group of EPDevice sources.

New data item requires configuration of the following properties:

- Type
- Value sources
- Value format
- Value buffer
- Conditions

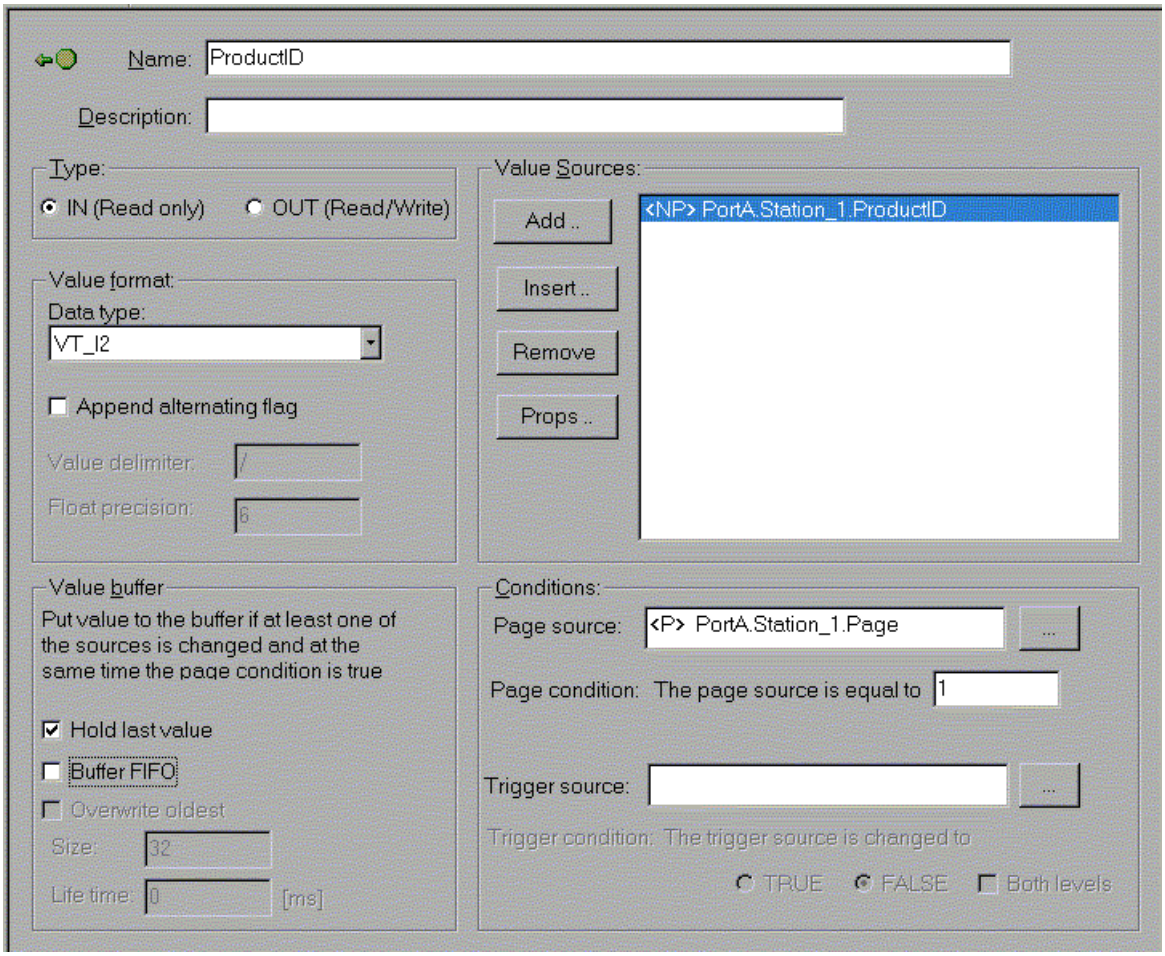


Figure 29 – The 'Data items' configuration table

8.3.2.0.1. Type

The type defines access rights of the data item. It can be set to one of the following values:

- *IN (Read-only)*
- *OUT (Read/Write)*

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

8.3.2.0.2. Sources

Sources represent the EPDevice sources that create the value of the data item. These sources are selected from the address space of the communication driver that is defined in the parent device.

Use buttons *Add, Insert* and *Remove* to add or remove a source from the list. Use button *Props* to change the properties of the selected source. In the dialog box for setting of the properties you can change the source name and specify, when the source should be updated.

8.3.2.0.3. Source properties

It is possible to change the *Source ID* of each source from the source list and to specify the *Update source* property.

Sometimes it is not necessary to update sources permanently, but only when a specific condition is true (a certain key is pressed, a certain page is activated...). This behavior is specified by the *Update source* property. This property can be set to one of the following values:

- If trigger condition is true
- If page condition is true
- Always

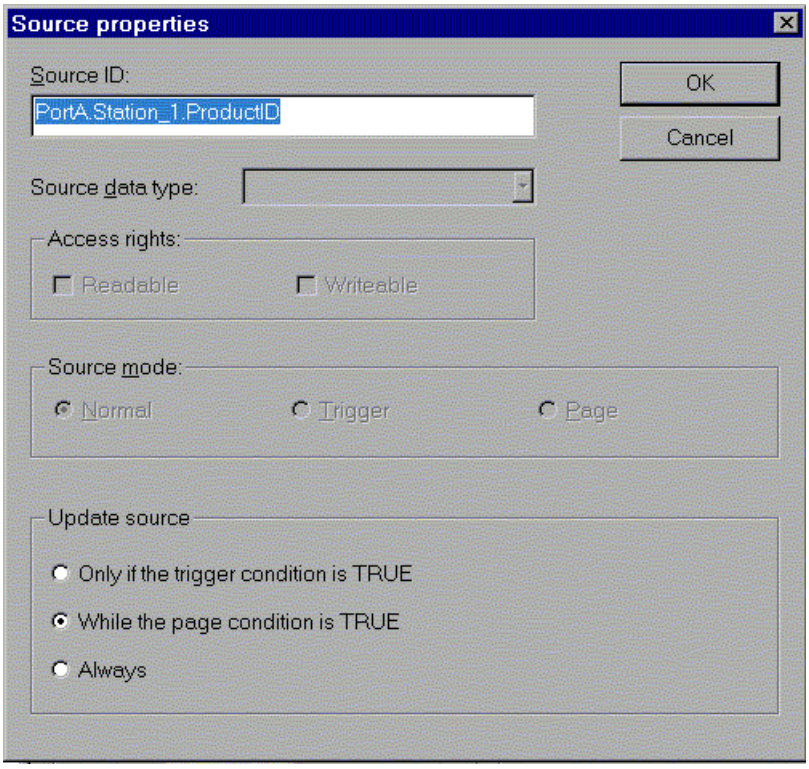


Figure 30 – The source properties

If the *Update source* property is set to *If the trigger condition is TRUE*, then the source is updated only if the trigger condition is true. For example a source can be updated only when the user of the EPDevice presses a certain key.

If the *Update source* property is set to *While the page condition is TRUE*, then the source is updated periodically while the page condition is true. If the page condition is false, then the source is not updated. This can be used for the sources that can be changed only when a certain page is active. For example sources in the EPDevices are mostly represented by input and output variables and are organized and displayed on more pages. But the user can edit (change) only those input variables, which are displayed on the current page. Other input variables can't be changed and it is not necessary to update them.

The *Update source* property of the sources that can be changed anytime should be set to *Always.* An example of such source is a keyboard field, which represents the current state of the EPDevice keyboard. Similarly the sources that are specified in the EPDevice application to be updated permanently (see SPSPlus Win documentation) should have the *Update source* property set to *Always*.

The update period of always-updated sources is defined for each device by the update rate parameter.

The *Update source* property is indicated with *<NT>, <NP>* or *<NA>* at the beginning of the *SourceID* in the source list.

8.3.2.0.4. Value format

Definition of the value format of the data item requires setting of the following properties:

Data type	*The canonical data type of the data item.*
Append alternating flag	*If checked, an alternating flag is appended to the value.*
Value delimiter	*This character delimits values of different sources (only for VT_BSTR).*
Precision	*The precision of float values (only for VT_BSTR).*

The allowed data type of the value depends on the number of sources in the source list. If there is only one source, then the data type can be one of available variant types. Usually it should correspond to the data type of the source. If a value should be prepared from more sources, then only following two data types are available:

VT_BSTR	***The values of the sources are converted to a string and linked to one string. The strings can be delimited with a defined character.***
VT_ARRAY| VT_VARIANT	***The values of the sources are placed in an array of variants.***

An alternating flag can be appended to each value. Its value will alternate from 1 to 0 whenever a new value is created and inserted to the buffer. The value of this flag is always placed at the end of the data item value.

If the data item groups more sources, then the value written by the client should contain values for each of the defined sources (in the specified string or variant array format). Otherwise the write operation will fail.

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

8.3.2.0.5. Conditions

The conditions are used to control when the value sources should be updated and when a new value of the data item should be created from the updated sources and put into the buffer. Properly defined conditions can minimize the transfer on the communication line between the server and the EPDevice. It is possible to use no conditions and update sources of all data items permanently. But in this case maybe it would be better to use only the communication driver (e.g. EPModbus).

For each data item it is possible to specify two conditions:

- Trigger condition
- Page condition

The **page condition** is defined by a page source and a page value. If the value of the page source is equal to the defined value, then the page condition is true. Otherwise it is false.

The **trigger condition** is defined by a trigger source and a trigger value. If the value of the trigger source is changed to the defined value, then the trigger condition is true. Otherwise it is false. Notice that the trigger condition is true only when the trigger source is changed. After the value of the condition is used, then it is automatically reset to FALSE.

The page and trigger sources should be selected from the address space of the communication driver. It is not necessary to define both conditions. In some cases the conditions are not necessary.

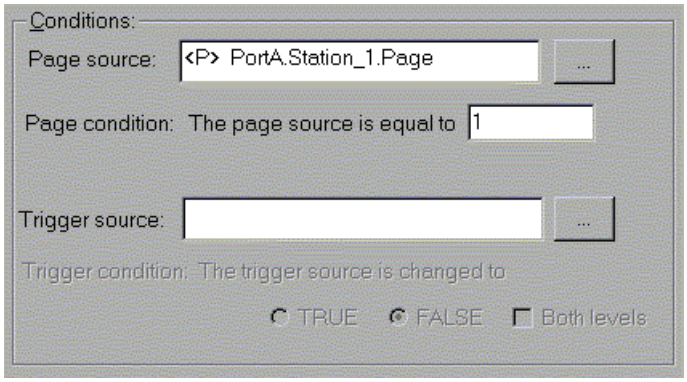


Figure 31 – The conditions

For example a variable ProductID is defined in the EPDevice. This ProductID is connected to an input box object placed on a page number 2. This variable can be changed only when the user selects page 2 and types new number in the input box. If we want to update this source only when the page 2 is selected, then we need to define a page condition. As a page source should be selected the source which was designed in the EPDevice to store the current page number. The page number in the page condition should be set to 2.

Finally the update source property of the ProductID source in the source list should be set to *While page condition is TRUE.*

After the value sources are updated, the same conditions are used to determine whether a new data item value should be created from the sources and put to the buffer or not. The rules are following:

Rule 1:

If no trigger condition is defined then a new data item value is created and put into the buffer only if at least one of the updated sources was changed. If a trigger condition is defined than the new value is prepared and put into the buffer only if this condition is true.

Rule 2:

If the page condition is defined, then a new value is created and put to the buffer only if this condition is true.

For example it may happen that a page condition is false and a trigger condition is true at the same time. In this case no value is created. It can be created only if both conditions are true.

8.3.2.0.6. Value buffer

A client program can specify the highest rate at which it is able to receive data from the EPCom server. But data item value can be changed faster. It usually corresponds to the update rate of the data item sources. This is specified by the *Update rate* parameter that is defined for each device in the server address space.

The changed data item values, which are not processed immediately by the client, are queued in a buffer. The size of the buffer is configurable. The buffer works like FIFO queue– First In First Out. This buffer enables clients to receive every change of the data item value.

The definition of the buffer requires setting of the following parameters:

Hold Last	*If checked, the last value of the data item is hold in the buffer.* *If unchecked, the last value of the data item is hold in the buffer but the value quality is changed to uncertain after the client reads it.*
Buffer FIFO	*If unchecked, the size of the buffer is default 1.* *If checked, it is possible to specify the size of the buffer.*
Overwrite oldest	*If checked, the oldest value placed in the buffer queue will be overwritten when the queue is full.*
Size	*The maximum number of values allowed in the buffer queue.*
Life time	*The lifetime of values in the buffer (not supported).*

Notice that the value buffering works only if the data item is active. Inactive items are not updated at all.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

9. Exicom/ProVicom OPC server administrators

There can be created more configuration databases for both EPCom and EPModbus OPC servers, but only one can be active. To view and to change the active configuration database following administrators can be used:

- **EPModbus Administrator**
- **EPCom Administrator**

The Administrator is a simple application that allows viewing or setting of the active configuration file of the server. It stores also a list of recently used configurations and it enables switching between several often-used configurations.

It is possible to change the current configuration database directly from the configurator using Make Active command. The administrator should be used mostly for security purposes - it can not change the contents of the configuration database (as it is possible in the configuration module).

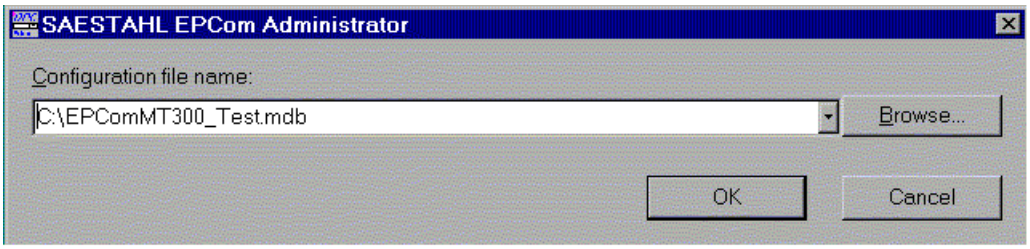


Figure 32 – The EPCom administrator

10. Configuration using SPSPlusWin

The configuration databases of the EPModbus and EPCom OPC servers can be also generated in the environment of the SPSPlusWin engineering tool.

10.1. Step by step procedure

- **Select E/P OPC server target device.**

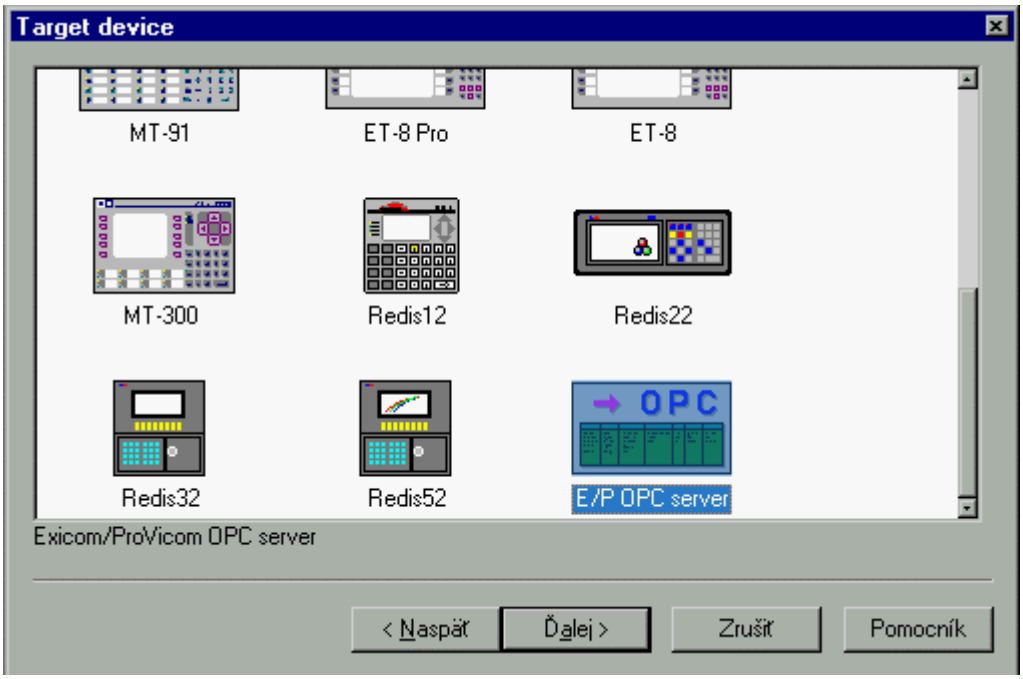


Figure 33 Target device dialog

- **For each used communication port define driver OPC Modbus.**

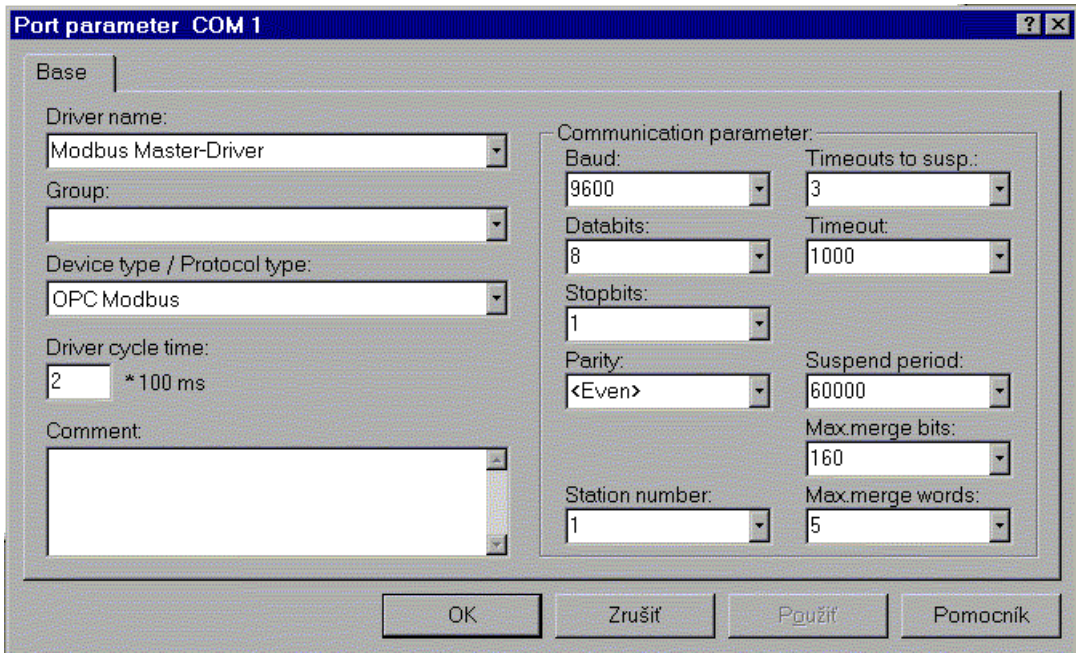


Figure 34 The Port parameters dialog

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

There are standard parameters such as Baudrate, Databits, Stopbits, Parity, and StationNo.

The meaning of the other parameters of the OPC Modbus driver is following:

Timeout - read/ write timeout, period length the driver will wait for response.

TimeoutsToSuspend - the number of subsequent timeouts to suspend a station from the pool cycle.

SuspendPeriod - the time interval the station is kept in suspended state

Max. merge bits - maximum block length of unused adjacent bits that can be transferred in one message.

Max. merge words - maximum block length of unused adjacent registers that can be transferred in one message.

- **For each communication port define connected stations.**

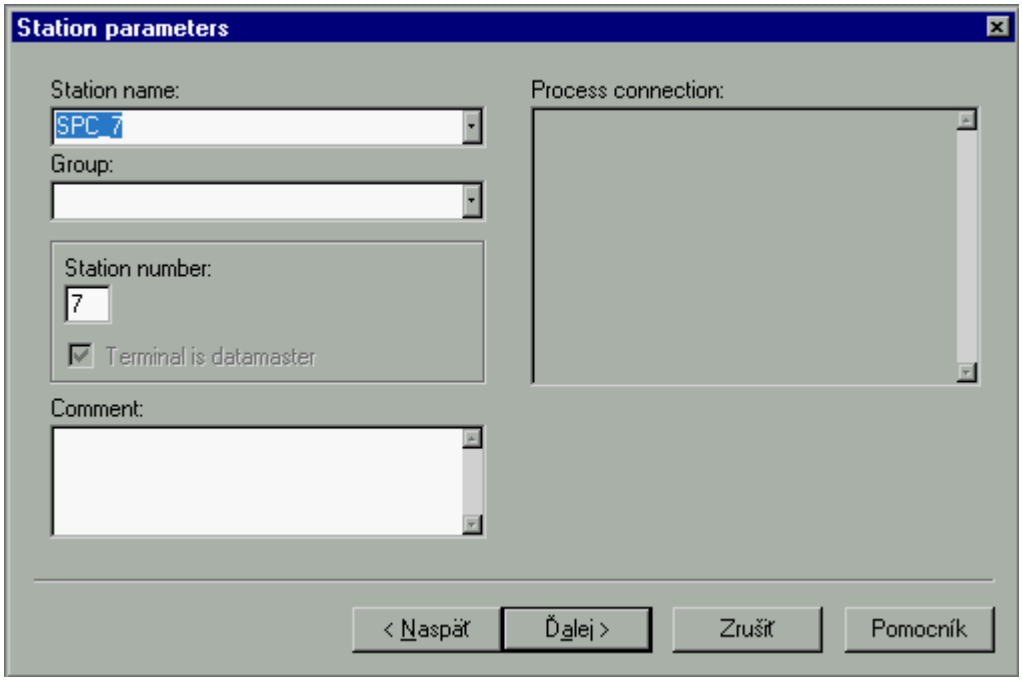


Figure 35 The station parameters

The field ‘Station name’ should contain the name of an existing SPSPlusWin project that defines the connected station. The modbus address of the station should be defined in the field ‘Station number’.

- **From the menu ‘Project’ select function ‘Generate’**

SPSPlusWin creates two configuration databases EPCom.mdb and EPModbus.mdb. These databases are stored in the current project directory (SPSPlusWin\Projects\ProjectName).

10.2. The hierarchy of the SPSPlusWin projects

The Project database of the device E/P OPC server defines the driver and connected Exicom/ProVicom devices. This project database is the **root database**. Each connected Exicom/ProVicom device is defined in a stand-alone project database. This database defines application of the device (pages, variables, process connections...). This project database is the **leaf database**. More than one device can be connected to the server – more than one leaf database can be linked to the root database.

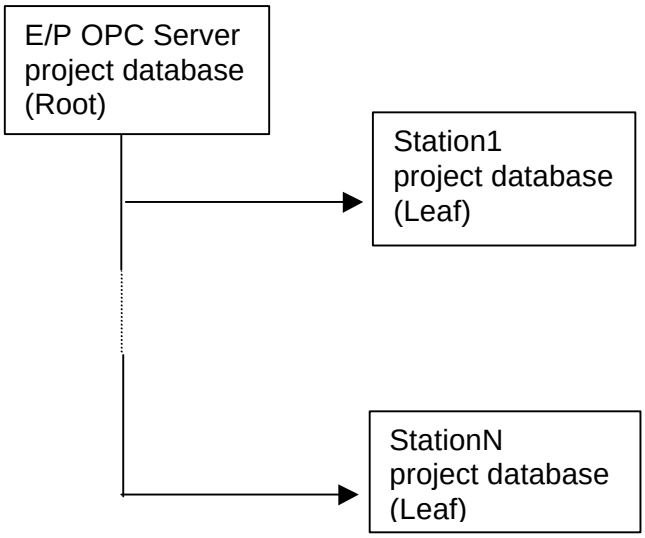


Figure 36 The hierarchy of the SPSPlusWin databases

10.2.1. The root database

This database defines:
- Communication drivers on selected communication channel.
- Exicom/ProVicom devices connected to the communication drivers

The OPC server supports only OPC_MODBUS driver. This driver can be selected for each of the four communication ports.

For each communication port can be defined up to 247 devices (stations). Each device is defined by the Station address and the name of the SPSPlusWin project database that defines the Exicom/ProVicom device (see Figure 35).

10.2.2. The leaf database

The leaf database defines the application of the Exicom/ProVicom device. The application is complex set of pages, messages, variables, process connections and drivers. Exicom/ProVicom device can use multiple drivers at the same time (for example it can communicate with PLC on one channel and with OPC server on another channel).

As mentioned above, the root database has reference to the leaf database (Station address and name of the project database). This database is valid only if it defines driver MODBUS_SLAVE with parameter **StationNo** equal to the requested **Station address (**defined for the device in the root database). For this driver should be defined only one station with the parameter **Station address** equal to the **StationNo** parameter of the driver OPC_MODBUS defined in the root database (see Figure 37).

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

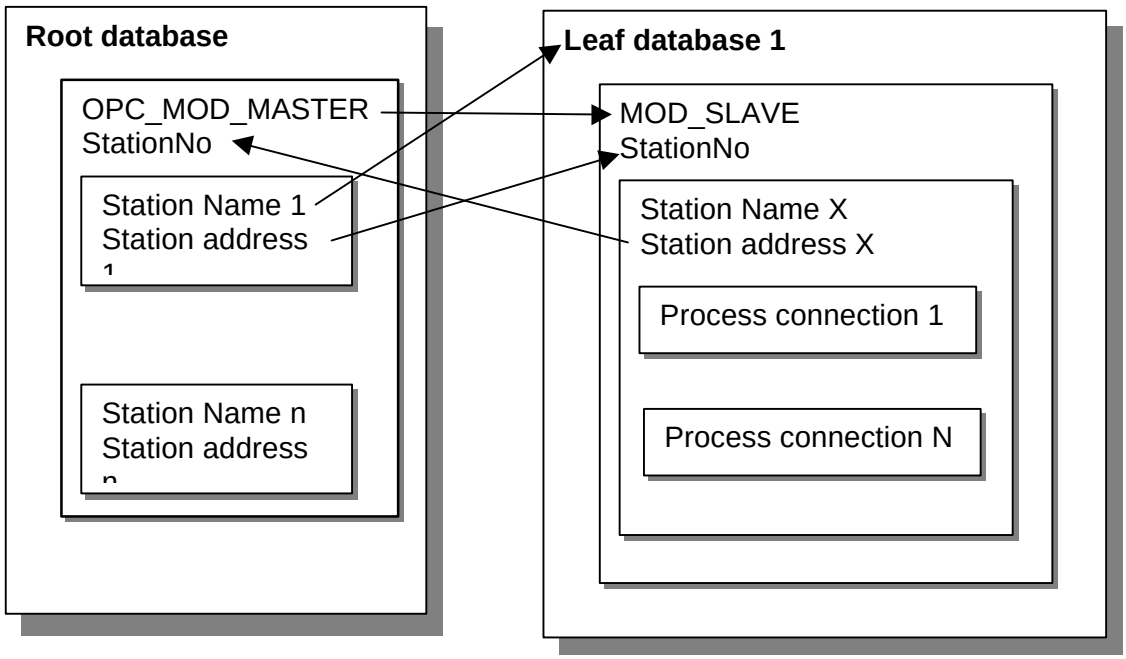


Figure 37 The connection between root and leaf databases

10.2.3. The generated configuration databases

SPSPlusWin generates two configuration databases. One for EPModbus OPC server. One for EPCom OPC server. The generator explores the root and leaf databases and creates both configuration databases accordingly.

EPModbus
The generator generates the address space of the EPModbus server – PORTS, DEVICES and DATAITEMS.

For each communication channel defined in the root database is created one item in the PORTS table. For each device connected to the port and defined in the root database is created one item in the DEVICEs table.

For each valid process connection (MODBUS_SLAVE driver with the proper StationNo) defined in the leaf database is created one item in the DATAITEMS table. Also, for each device are generated two data items for diagnostic purposes – PortState and DeviceState.

EPCom
The generator generates the address space of the EPCom server – DEVICES and DATAITEMS.

For each device defined in the root database is created one item in the DEVICE table.

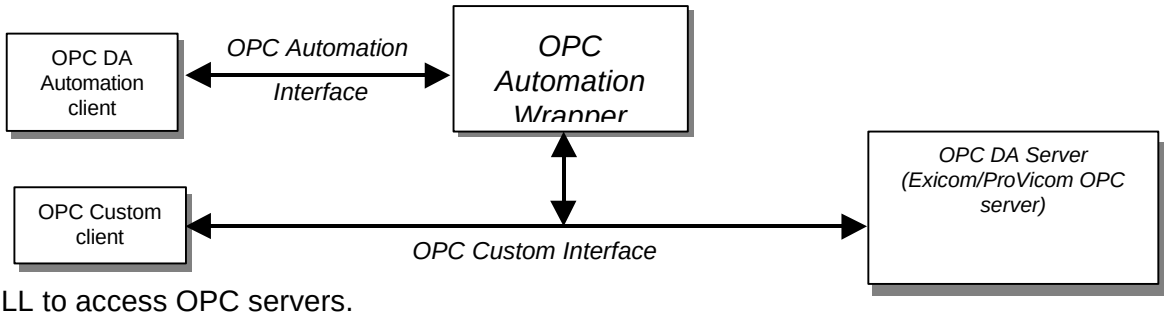
For each variable with valid process connection is created one item in the DATAITEMS table. The data items are organised according to the type of the variables:
- analogs
- digitals
- strings
- error messages
- operation messages
- recipes
- station bit fields
- diagnostic

11. OPC Clients

This chapter describes two simple OPC client programs:

- OPC Custom client
- OPC DA Automation client

The OPC Custom client can communicate with any OPC DA Server through the OPC custom interface. The OPC DA Automation client uses the OPC DA Automation Wrapper



DLL to access OPC servers.

Figure 38 – The OPC Custom and automation clients

11.1. OPC Custom client

This client application is a very simple client for testing of connection with any OPC server through OPC DA custom interface. It is an MFC single document interface application.

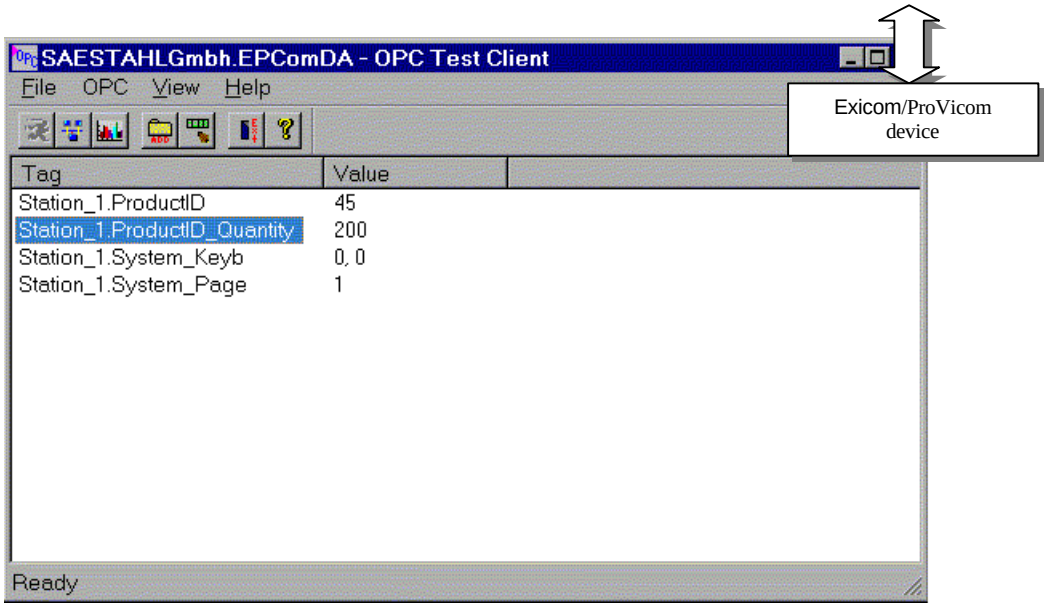


Figure 39 – The OPC Test Client

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

11.1.1. How to connect to a server?

Select ***Connect*** command from the OPC menu or press the ***Connect*** button on the toolbar.

Choose one server from the list of available local OPC servers and press OK button.

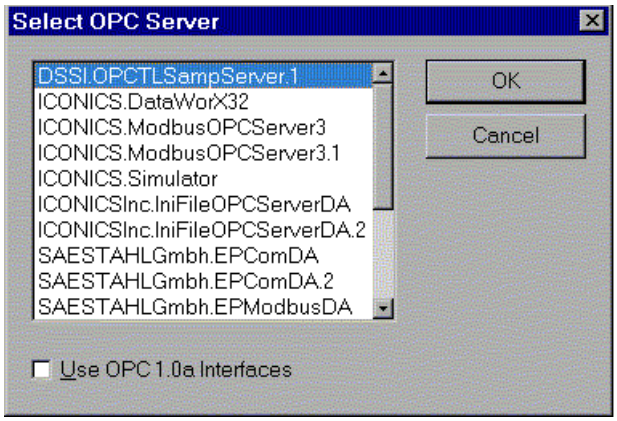


Figure 40 – The list of local servers

To disconnect from the server select ***Disconnect*** command from the OPC menu or press the ***Disconnect*** button on the toolbar.

11.1.2. How to display server status information?

Select ***Server status*** command from the OPC menu or press the ***Status*** button on the toolbar.

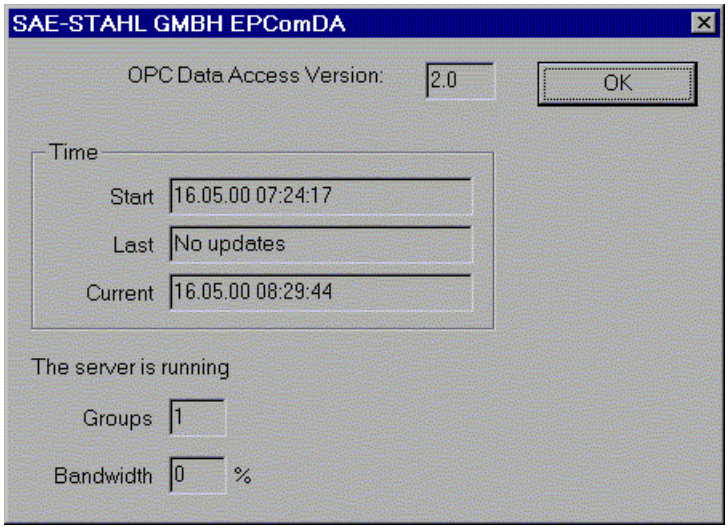


Figure 41 – The server status information

11.1.3. How to change group parameters

After the server is connected, the client program will automatically create one private group in the server. It is possible to change some of the group parameters.

To change these parameters select ***Group parameters*** command from the OPC menu.

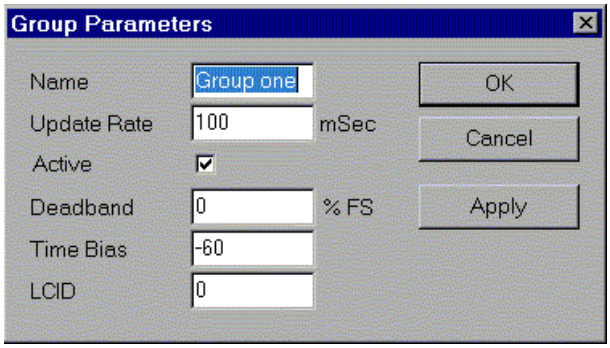


Figure 42 – The group parameters

The meaning of the group parameters is following:

- *Update rate* represents the period of updating of the items in the server.
- The group can be *active* or *inactive*. If it is Inactive, the items in the group are not updated at all.
- *Deadband* – server sends notify messages only when a value of an item is changed more than specified deadband. This parameter is in percent of the full scale (FS) of the item source.
- Using *TimeBias* the time base of the server time can be shifted (in minutes).
- *LCID* is a localization ID of error strings.

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

11.1.4. How to add a new item to the server group

Select ***Add item*** command from the OPC menu or press the ***Add item*** button on the toolbar.

Select required ***Data Type*** of the items that should be displayed.

Use the left pane to browse in the server address space.

Select required item from the right pane and press OK button.

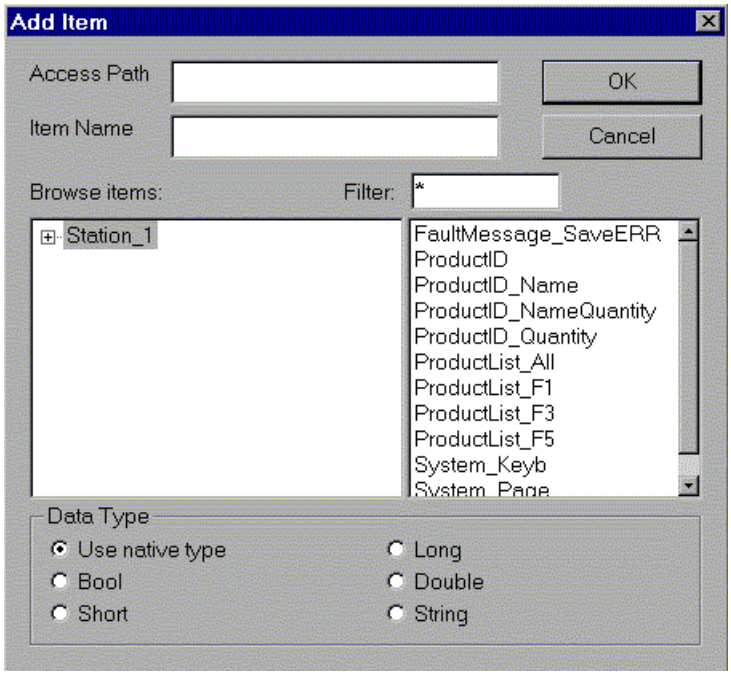


Figure 43 – The dialog for browsing of the address space

11.1.5. How to write a value to the item

Click (select) the required item in the displayed list.
Select ***Write value to item*** command from the OPC menu or press the ***Write item*** button on the toolbar.
Type a new value and press the OK button.

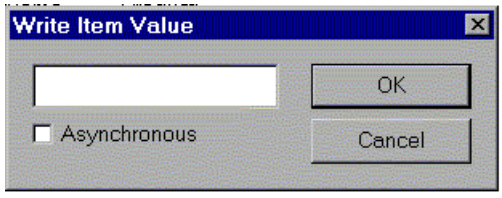


Figure 44 – The dialog for writing a new value to an item

11.2. OPC DA Automation client

This client application is a simple client for testing a connection with any OPC server using OPC DA Automation wrapper DLL. It also demonstrates on a simple example how to use the Exicom/ProVicom OPC server in VBA application. To run this client you will need:

- Exicom/ProVicom device (MT/ET_xxx) connected to your PC
- MS Excel 97
- SPSPlus Win v.X.X.X

The OPC automation client is created in MS Excel and is implemented in the workbook OPCExampleClient. This workbook contains following sheets:

- List of OPC DA servers
- Server status
- Server items
- Product list

The first three sheets can be used to test the connection with any OPC server. The last sheet demonstrates the use of the Exicom/ProVicom OPC server.

11.2.1. How to run the OPCExampleClient?

Follow these steps:

1. Use SPSPlus Win to download the MTxxx_OPCExample application to the target device (MTxxx). This application should be stored in the MTxxx_OPCExample.zip archive in the SPSPlusWin\Projects directory.
2. Use EPModbus Administrator to activate EPModbus_MTxxx_OPCExample configuration database. This should be stored in the directory ..\SAE-STAHL\Exicom-ProvicomOPCServer\EPModbus\Data.
3. Use EPCom Administrator to activate EPCom_MTxxx_OPCExample configuration database. This should be stored in the directory ..\SAE-STAHL\Exicom-Provicom OPC Server\EPCom\Data.
4. Select the OPC DA Automation Client from the start menu. This operation should start MS Excel and open the OPCExample workbook.

 © SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

11.2.2. Sheet1 – List of OPC DA servers

After this sheet is activated, it will display a list of all OPC servers available on this computer.

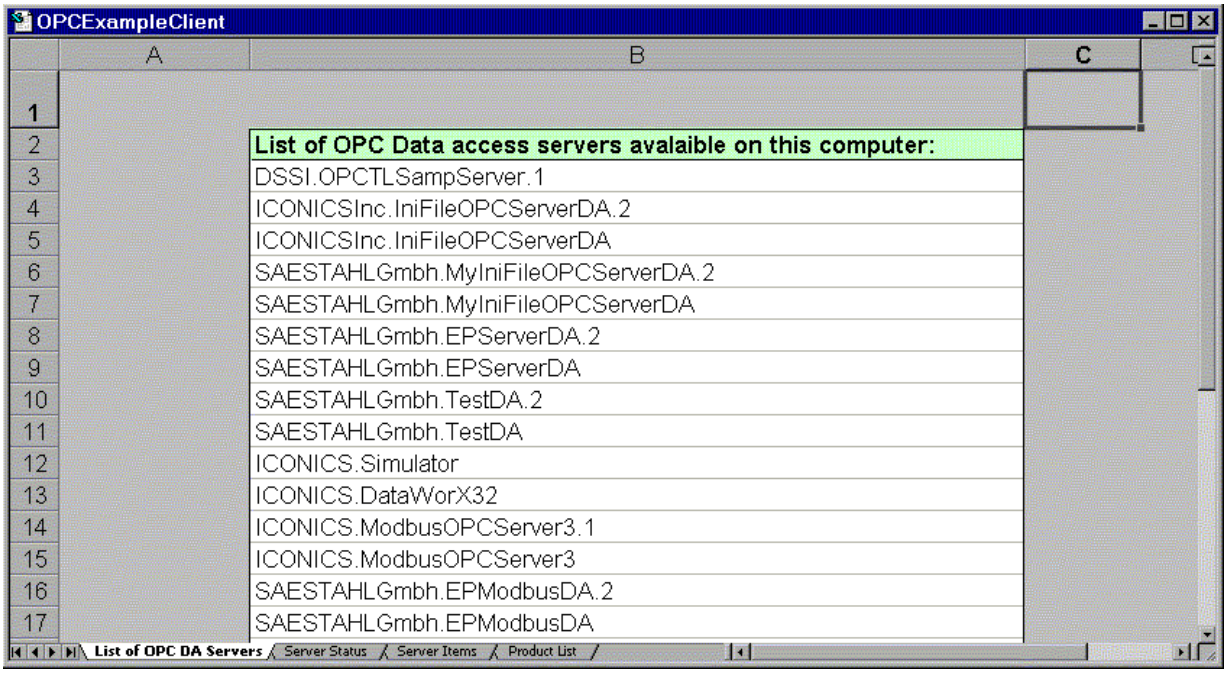


Figure 45 – 'List of OPC DA Servers' sheet

Notice that some of the servers have two different ProgIDs (programmatic identifiers). For example SAESTAHLGmbh.EPModbus and SAESTAHLGmbh.EPModbus.2. The first one is a version independent ProgID, the second one is a version depended ProgID.

11.2.3. Sheet2 – Server status

After this sheet is activated, it is connected to the server specified in the C2 cell and the current status information of the server is displayed.

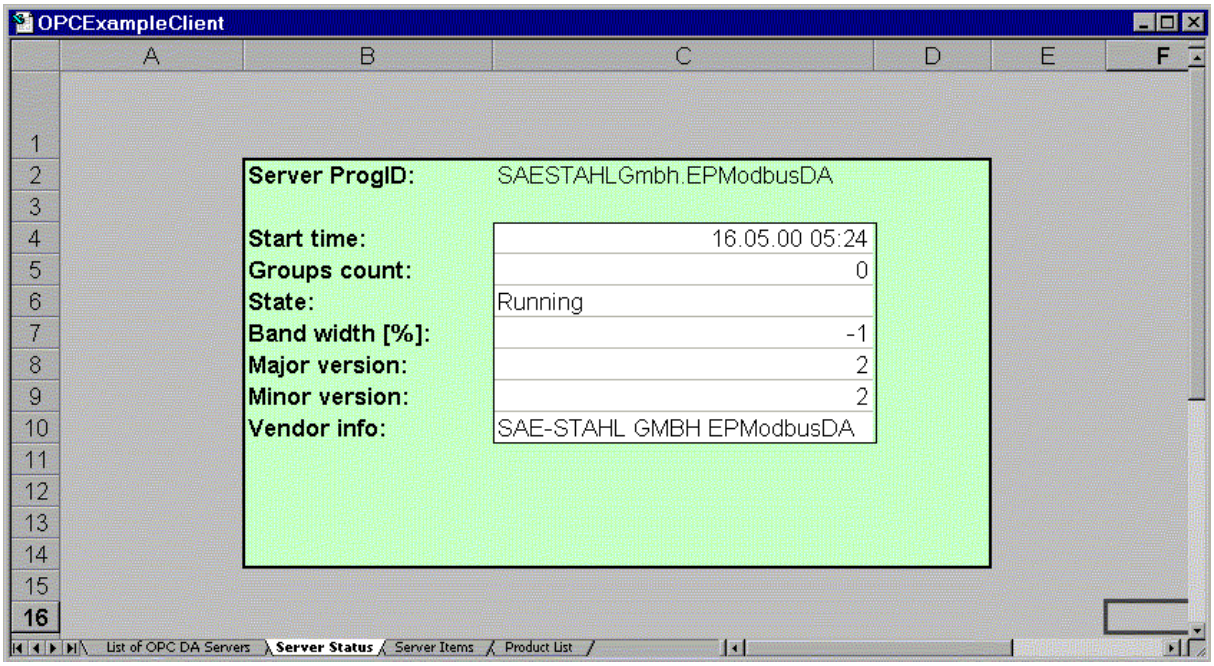


Figure 46 – 'Server status' sheet

11.2.4. Sheet3 – Server items

After this sheet is activated, it is connected to the server specified in the C2 cell and the first ten items from the server address space are monitored – current value, quality, time stamp, data type, active state, client and server handles.

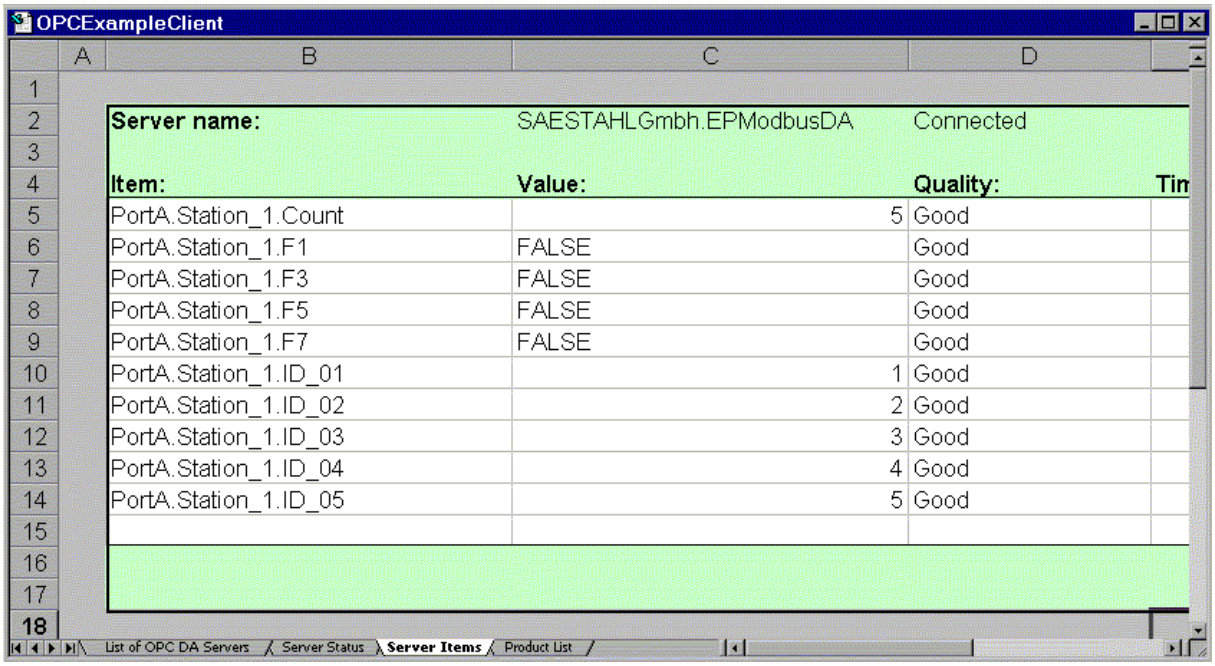


Figure 47 – 'Server items' sheet

© SAE-STAHL GMBH / OPC_GB.DOC / 24.01.01

11.2.5. Sheet4 – Viewing and editing the product list

This example demonstrates situation when Exicom/ProVicom device is used to access a database placed on a connected PC. When user types a ProductID number in the device, the Exicom/ProVicom OPC server should notify a client program. The client program should read data from the database according to the ProductID and send them to the device.

A table of products represents the database. Each record consists of a Product ID, Name and Quantity fields. All client actions are implemented in a simple VBA script distributed with this workbook.

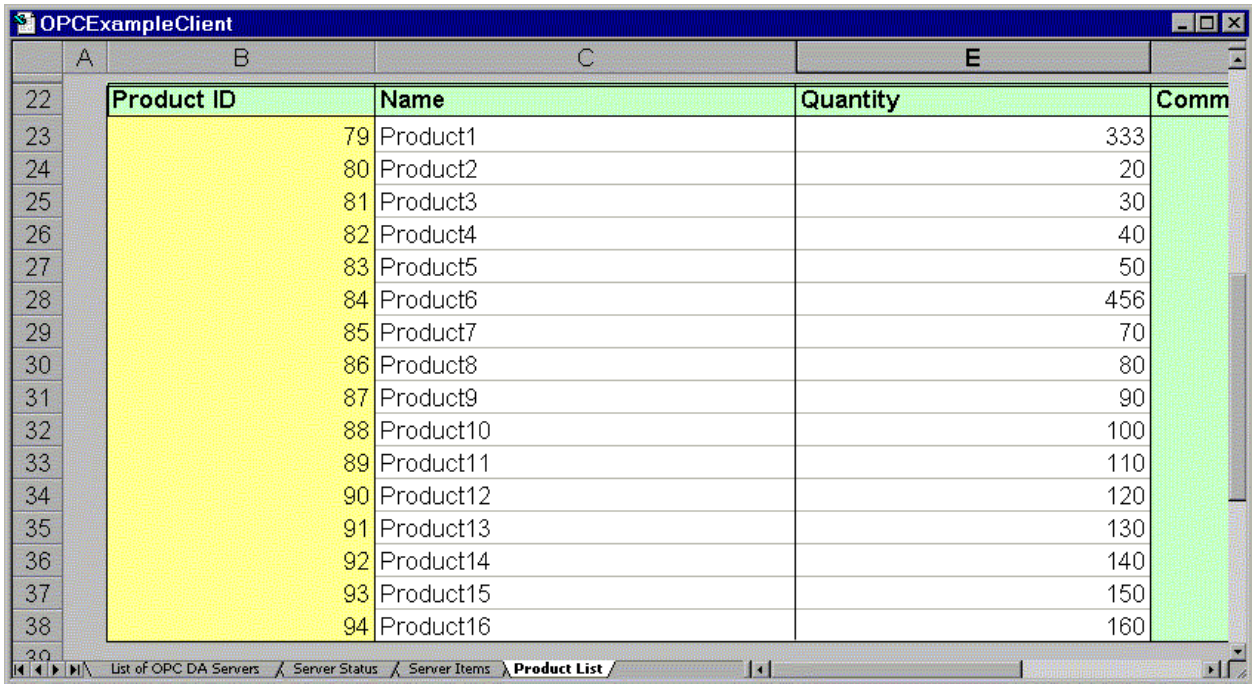
This client is implemented in the sheet ProductList of the OPCExampleClient workbook.

There are two possibilities how to access data from the database in this device application:

- Using Product ID page
- Using ProductList page

ProductID page:
Type one of the product identification numbers in the ‚Product ID' input box. After the 'enter' key is pressed then a corresponding name and quantity should be displayed.

ProductList page:
Use F1 and F3 to display previous or next table records. Use arrows to go through input boxes in the table. Use F5 button to save the changed records.



	Product ID	Name	Quantity	Comm
23	79	Product1	333	
24	80	Product2	20	
25	81	Product3	30	
26	82	Product4	40	
27	83	Product5	50	
28	84	Product6	456	
29	85	Product7	70	
30	86	Product8	80	
31	87	Product9	90	
32	88	Product10	100	
33	89	Product11	110	
34	90	Product12	120	
35	91	Product13	130	
36	92	Product14	140	
37	93	Product15	150	
38	94	Product16	160	

Figure 48 – 'Product List' sheet